Theorist®

reference manual

This manual reflects Theorist Version 1.10 software.

Prescience Corporation

Copyright © 1990 by Prescience Corporation. All rights reserved. No part of this work covered by the copyrights hereon may be reproduced or copied in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or information retrieval systems—without prior written permission of the publisher:

Prescience Corporation
939 Howard Street, #306
San Francisco, CA 94103

First Printing: October 1990
Second Printing: January 1991

Theorist and Expressionist are registered trademarks and Projectionist is a trademark of Prescience Corporation.

Apple, Finder, ImageWriter, LaserWriter, Macintosh, MultiFinder, and QuickDraw are trademarks of Apple Computer, Inc.

PostScript is a trademark of Adobe Systems, Inc.

Other names and products are trademarks of their respective owners.

Contents

Preface ix
Who This Manual is For xi
What This Manual is About xii

Introduction 1
System Requirements 4
Algorithms and Numeric Precision 5
Mathematical Perfection 7
Importing and Exporting Data and Graphics 9
Printing 10
Propositional Schema 12

Notebooks 15
Notebook Files 18
Startup Notebooks 19
Stationery Files 19
Managing Notebooks 20
Distribution Notebooks 21

Propositions 23
Statements 26
Working Statements 28
Value Statement 29
Maximum Statement 30
Minimum Statement 30
Increment Statement 30
Function Statement 31
Order Statement 31
Generating Case Theories 32
Creating Case Theories 33
Graph Theories 36
Comments 37
Name Declarations 39
Independence Declarations 42
Transformation Rules 45
Outlining 47

Expressions....49
Class....52
Constants....53
Variables....53
M-Linear Operators....54
D-Linear Operators....54
Functions....55
Numbers....57
Names....59
Subscripted Names....60
Predefined Names....62
Ops....69
Addition....71
Negation....71
Subtraction....71
Multiplication....71
Division....72
Equals....72
Relational....72
Function....73
Power....73
Index or Subscript....74
Absolute Value....74
Factorial....74
Square Root....75
Adjoint....75
Matrix....75
Dot Product....76
Cross Product....77
Partial Derivative....77
Integral....78
Summation....78
Pi Product....79
Conditional....79
Evaluate At....80
Range....80
Wildcard Variables....81
Collapsars....83
Parentheses....84
Question Mark....85

Manipulations 87
Manipulation Options 90
Auto Simplify 90
Auto Casing 91
Arbitrary Constants 91
Manipulating in Place 92
Hand (Mouse) Manipulations 93
Isolate 93
Move Over 96
Commute 96
Substitute 98
Command Manipulations 101
Calculate 101
UnCalculate 104
Simplify 105
Expand and MiniExpand 109
Collect 112
Factor 114
Apply 116
Other Manipulations 120
Transform 120
Taylor Series 122
Integrate by Parts 125

Graphs 129
Graph Theories 132
Creating Graph Theories 136
Two-Dimensional Graphs 162
Editing Icons 142
Graph Details 144
Creating 2D Graphs 148
2D Graphs 152
Finding Roots 159
Three-Dimensional Graphs 162
Editing Icons 164
Graph Details 166
Creating 3D Graphs 171
3D Graphs 172
Surface Plot Propositions 181
Color Schemes 183

Graphs (continued)
Animation....193
Printing and Exporting Graphs....198
Graph Preferences....199
Printing Graphs....202
Exporting Graph Images....202
Exporting Graph Data....203

Graph Programming....207
Overview....210
Comparing Graph Details....211
Creating a Graph from Scratch....215
Reordering Dependencies....217
Spherical Contour Plots....219

Editing....221
The Structure of Expressions....224
Editing Basics....225
Precedence Hierarchy and Fortranish....226
Escape Levels....230
Using the Palette....231
Example Sequence of Deletions....234
Selecting and Editing Objects....234
Selecting and Editing Propositions....237
Selecting and Editing Expressions....237
Selecting N-ary Expressions....238
Selecting Names and Numbers....239
Selecting Matrices....239
Selecting Comments....241
Examples of Expression Creation and Modification....242
Simple Expressions....243
Functions....252
Linear Operators....253
Matrices....254
Differential & Integral Expressions....255
Iterations....257
Other Expressions....258

Tips & Techniques....259
Algebra....262
Moving Negations....262
Moving Factors....262
Making a Real Variable....262
Making a Variable a Constant....262
Enclosing Expressions....263
Using Substitutions....263
Factoring multivariate polynomials....264
Graphs....266
Subscripted variables....266
Balancing Bounds....266
Rotating an Animation....266
Editing....267
Negating Expressions....267
Editing on the Left....267
Editing Matrices....267
Creating Identity Matrices....268

Menus....269
Menu Commands....271
File....272
Edit....274
Input....276
Notebook....278
Manipulate....280
Graph....282
Prefs....285

Appendices....287
Appendix A: Keyboard Maps....289
Appendix B: Using Projectionist....299
Appendix C: Using Expressionist with Theorist....305
Appendix D: Distribution List....306

Glossary....309

Bibliography....325

Bug Report Form....330

Index....333

Preface

Who This Manual is For

The *Theorist Reference Manual* provides a complete description of Theorist, the Macintosh software package for algebraic manipulation and visualization.

Theorist runs on the Macintosh and this manual assumes that you are familiar with this computer, or have access to the *Macintosh Reference* manual. That manual, available from Apple Computer, provides step-by-step instructions for all of the basic tasks you can perform with a Macintosh.

This manual also assumes that you have a basic understanding of mathematics and want to use Theorist to pursue scientific or engineering research, or are studying mathematics, and want to improve your grasp of mathematical relationships in your field. This reference manual is not designed to teach mathematics but to describe how Theorist works so that you can explore, and gain new insight and understanding in any mathematical subject.

The basic structure of this manual follows the logical structure of Theorist from the outside in, from the broadest conceptual categories to the finest. For an instructional introduction to Theorist, see the *Theorist Learning Guide*.

What This Manual is About

The *Theorist Reference Manual* contains ten chapters, four appendices, a glossary, a bibliography, and an index.

Introduction
Discusses system requirements, and describes Theorist's underlying concepts, program algorithms, importing and exporting data and graphics, and printing.

Notebooks
Describes Theorist's files.

Propositions
Describes the entries you can make in a notebook.

Expressions
Describes the fundamental units used to create propositions.

Manipulations
Describes how to algebraically manipulate expressions and equations.

Graphs
Describes the two- and three-dimensional graphs you can create.

Graph Programming
Describes how to create graphs of your own design.

Editing
Describes how to edit mathematical expressions in Theorist.

Tips and Techniques
Provides a few tips for improving your speed and efficiency.

Menus
Lists all menu commands and provides a brief description of each.

Appendix A: Keyboard Maps
Provides keyboard maps for Command- and Option-key combinations.

Appendix B: Using Projectionist
Describes how to use Projectionist, a basic program provided with Theorist for viewing graphic animations.

Appendix C: Using Expressionist with Theorist
Describes how to move expressions between Theorist and Expressionist, mathematical typesetting software from Prescience.

Appendix D: Distribution List
Lists the contents of the Theorist distribution disks.

Glossary
Provides brief descriptions of key words used in this manual.

Bibliography
Lists some books useful for exploring computer mathematics in greater depth and detail.

Index
Cross-references all important topics in this manual.

Introduction

Theorist is a software system designed for algebraic analysis, graphical visualization, and mathematical insight and understanding. This chapter introduces the program, and presents a discussion of:

- System requirements
- Algorithms and numeric precision
- Mathematical perfection
- Importing and exporting data and graphics
- Printing
- The Propositional Schema

System Requirements

The minimum configuration for using Theorist is:

- A Macintosh Plus or Classic (or larger machine)
- One megabyte of memory (or more)
- System 4.1 (or later system software)

Theorist runs on all Macintosh computers. The Mac II version (included as a separate program on the distribution disks) runs on the Mac II, Mac IIx, Mac IIcx, Mac IIci, Mac IIfx, Mac SE/30 or any Macintosh equipped with a 68020, 68030, or compatible CPU, *and* a 68881, 68882 or compatible floating point co-processor chip. Creating graphs (especially high resolution color plots) and manipulating expressions is measurably faster than the normal version.

The Mac II version takes advantage of the architecture of both chips for increased performance. Therefore, it does *not* run on 68000 machines with add-on math co-processors nor 68000 machines with 68020 accelerators that lack a math co-processor (68881 or 68882).

Except for increased speed of operation, both versions are essentially identical. The 68881 floating point co-processor yields only about 15 digits of precision for some transcendental functions, instead of the usual 19 digits. Certain algorithms take this into account in the Mac II version to increase performance. If you need this extra precision, use the regular version.

Mathematical manipulations are known to be compute intensive. The more memory (RAM) your machine has, the faster Theorist operates. Pay attention to memory usage—if Theorist runs out of memory, it may be forced to quit. A memory "thermometer" is displayed along the top of the entry palette so you can see how much memory is in use.

If you receive a message that says you are inexplicably running low of memory (or have run out of memory), save your work, quit the program, and start again. These messages can indicate that some memory may no longer be available; its best to free up all memory associated with the program and start again.

Algorithms and Numeric Precision

Theorist is designed to provide a superior work space for mathematical investigation while balancing conflicting demands for speed, accuracy, reliability, generality, and compact and efficient code. As the program evolves, different trade-offs will lead to the implementation of a modified set of algorithms.

All of the algorithms used in Theorist are either available to the general public in journal articles or are obvious to anyone skilled in software design. The same is true of all commercially available mathematical software, especially products that claim to use exotic, and by implication, superior, algorithms. Numerical analysis and symbolic algebra are far older than computer science, and computer graphics is a well established field of study. All three are developed technologies, and the best algorithms are well known.

This *Reference Manual*, and the *Theorist Learning Guide*, describe and explain the features of the interface—how you use the program—rather than the internal workings of the program. If you want to discuss the relative merits of mathematical algorithms, or have a problem with our implementation of an algorithm, write us a letter or give us a call. We live for mathematics and computers, and would be delighted to hear from you.

The program endeavors to return nineteen digits of precision for numeric calculations. In some instances, in order to avoid unreasonable processing time, some algorithms return less than nineteen accurate digits. However, the results should be sufficient for most purposes.

In particular, numeric integration usually returns values that are accurate to approximately nine digits. Results of the Gamma function and other special functions (e.g., Bessel functions) are accurate to approximately seven or eight digits. If you need to know the exact degree of numeric precision, check your results and solve equations in another way. Where possible, attack a problem from different directions, and compare the results.

Floating-point round off errors can also reduce the accuracy of calculations. All floating-point calculations have inherent limitations. For example, the following table shows what happens if you add and subtract a large number from a known value.

Expression	Calculated Value	Digits of Precision
$\frac{1}{3}$	0.3333333333333333333	19
$\frac{1}{3} + 1 \times 10^{15} - 1 \times 10^{15}$	0.33331298828125	4
$\frac{1}{3} + 1 \times 10^{16} - 1 \times 10^{16}$	0.3330078125	3
$\frac{1}{3} + 1 \times 10^{17} - 1 \times 10^{17}$	0.3359375	2
$\frac{1}{3} + 1 \times 10^{18} - 1 \times 10^{18}$	0.3125	1
$\frac{1}{3} + 1 \times 10^{19} - 1 \times 10^{19}$	0	0

Mathematical Perfection

Algebra is often a tricky business. Seemingly straightforward derivations can contain serious, and consequential, errors. One reason is that special cases can pop up when you least suspect them. Theorist handles special cases better than most computer algebra systems, but software cannot, by its nature, generate mathematical perfection. And even before computers were introduced, the validity of a mathematical derivation was often not an easy thing to prove.

There are a variety of situations where "trustworthy" manipulations give answers that are only true some of the time. Round off error can build up and overwhelm any significant result. A Taylor series, which is just a close approximation, is "equated" to the original function. If you change the behavior of names in the middle of a derivation, you can invalidate many of the manipulations already performed.

If you use Theorist with Auto Casing on, you soon realize that special cases can multiply like rabbits, especially for long derivations. Sometimes, in order to get anything done, you simply have to turn Auto Casing off.

For another example, $\sqrt{x^2}$ looks like it is equal to x. And this is the value that Simplify returns. But for real numbers, the solution is in fact $|x|$. For complex numbers the solution is even more subtle. Theorist returns the single value x, because the square root op is designed to return the principle value square root. In most cases this is entirely satisfactory, but there are exceptions.

Even in very simple situations, ambiguity can result. Consider the following equation:

$$x - y = 5 - y$$

The conclusion that $x = 5$ is valid—unless $y = \infty$. In that case, all bets are off.

There are myriad paths to mathematical misunderstanding. And computer-based systems add to this number their own special forms of possible error. No program can keep track of the many ways that you can go wrong and warn you of the dangers.

The moral of this story is that you should always check your work. For example, if you perform a symbolic integration, differentiate the result to confirm that your answer is correct. It is very seductive to imagine that your answers are always correct. It only take a few disasters to learn otherwise.

Importing and Exporting Data and Graphics

Theorist is designed to work cooperatively with other programs that run on the Macintosh. There are several ways to get numeric, textual, and graphic data into and out of the program. The following table lists the chapters where various importing and exporting activities are described.

To:	See:
Import data-point values	Editing chapter: Selecting Matrices
Import text or graphics (as Comments)	Propositions chapter: Comment propositions
Import mathematical expressions	Editing chapter: Fortranish and Appendix C: Using Expressionist with Theorist
Export data-point values	Graphs chapter: Copy Heights Array and Copy Points List
Export graphs (PICT or EPS)	Graphs chapter: Printing and Exporting Graphs

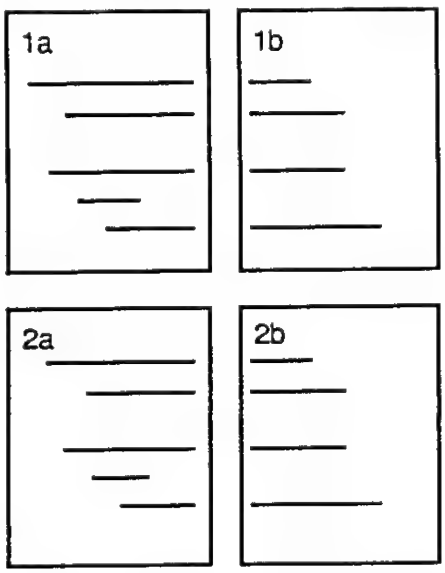
The graphs shown in the figures of this manual may look a bit different on your monitor. The appearance of a graph depends on the type of monitor that you are using. Theorist takes advantage of whatever display technology is available, black and white, gray scale, and all levels of color monitors.

Printing

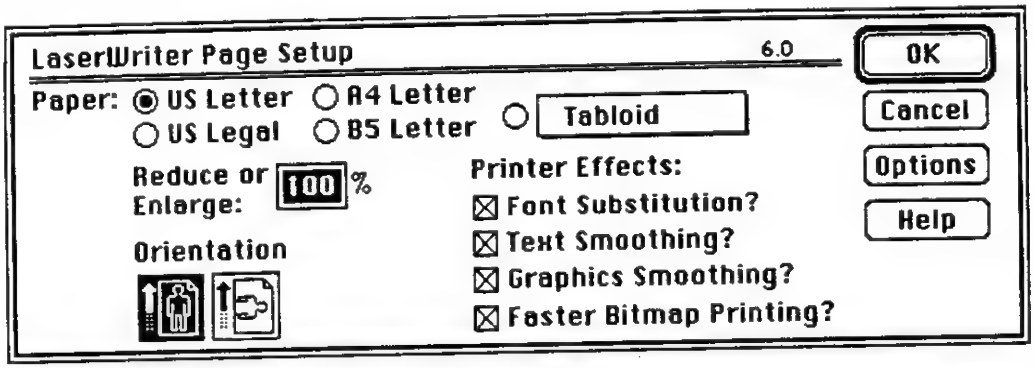
Theorist provides an environment for mathematical investigation, rather than a means to create formal presentation. To present your work, you can easily export expressions and graphs to page layout, graphics, and animation software packages.

For informal presentations and review, you can print your notebooks at any time: choose **Print** on the **File** menu (or press Command-P). A Theorist notebook is structured as an infinite two-dimensional surface. As such, notebooks do not always lend themselves to printing on fixed-size pieces of paper.

However, you can adjust several factors to produce a reasonable print out. After making these adjustments, if a notebook is still wider than one page, a dialog box comes up warning you that horizontal pages will be used. Pages are labeled to help you piece them together. The first row of pages is labeled 1a, 1b, 1c, and so on. The second row is labeled 2a, 2b, 2c, and so on. For example, the following figure displays four pages of a two-page wide notebook.



Choose **Page Setup** on the **File** menu to bring up a dialog box to set the print variables. This dialog box is different for different printers. If you are using a LaserWriter the dialog box lets you specify the size of paper to use, the percentage of image reduction, and the page orientation.



To save paper, if a notebook is four pages long but one wide expression extends to two pages horizontally, only one additional page is printed. That is, your printed notebook could have pages numbered 1a, 2a, 3a, 3b, 4a.

To help you determine in advance how many pages a notebook will require, small dotted tick marks (3 dots) are displayed at the top and bottom of each notebook window at the boundaries between horizontal pages.

If you are using the PrintMonitor under MultiFinder, some complex graphs may be larger than can be stored in the memory reserved for this utility. If this happens, a message is displayed declaring that the PrintMonitor does not have sufficient memory. To solve the problem, cancel any current printing jobs, select the PrintMonitor (in your System folder), choose **Get Info** from the **File** menu (or press Command-I), and increase its memory allocation.

Propositional Schema

The fundamental idea in Theorist is the proposition. A proposition is a mathematical statement of equality (or other relationship), that describes a particular ontological structure. Building from known or true propositions, you can derive conclusions that are also true. A theory consists of a group of propositions that are assumed to be true, and various conclusions that *are* true—if the original propositions are true. A theory can have many interrelated propositions, or just a few. For example, a simple theory could contain two equations:

$$x = 3$$

$$y = 2x + 5$$

By manipulating these two propositions, you can see that, in this theory, a new proposition stating that y equals eleven would be another true proposition. In another theory, where x is equal to five, such a proposition would not be a valid conclusion.

You use Theorist to enter propositions that you believe to be true. Then, by manipulating these propositions, you can derive other valid propositions. Theorist files, called "notebooks," consist of one or more hierarchically organized theories that contain propositions you enter, and valid conclusions derived from them.

Each proposition you enter is assumed to be true—in that theory. If you enter two contradictory propositions (e.g., $x = 3$ and $x = 5$), and try to derive a conclusion based on the value of x, Theorist marks which of the two proposition is used to derive the new conclusion. In effect, the program indicates that a derived proposition is true, only if *these* particular propositions are also true.

If you want to discover what valid conclusions can be drawn from different and contradictory propositions, you can easily branch off one or more "case theories" in which you explore situations where certain particular facts are true, as well as all propositions in any enclosing theory.

The propositional schema derives from the mathematical understanding that simple known facts of relationship can be combined to create dependable but as yet unknown truths. Though powerful, this scheme is susceptible to what is known to computer programmers as the problem of garbage in, garbage out (GIGO).

If you enter propositions that are contradictory, and manipulate those propositions explicitly, you can prove that which is obviously false. For example, you could enter the following propositions:

$x = 1$

$x = 0$

Now, if you combine these two propositions, by substituting the value of x established in the first equation into the second equation, your new "valid" conclusion is:

$1 = 0$

Clearly, this is not the way to Truth and Understanding. As you can see, if you put garbage into a theory, you're likely to get garbage out.

Notebooks

A Theorist notebook is a hierarchically organized work space for algebraic, numeric, and graphical study. Whenever you work with Theorist, you work in a notebook. Notebooks contain expressions, equations, theories, comments, declarations, rules, and two- and three-dimensional graphs. Each item in a notebook is either a proposition or part of a proposition.

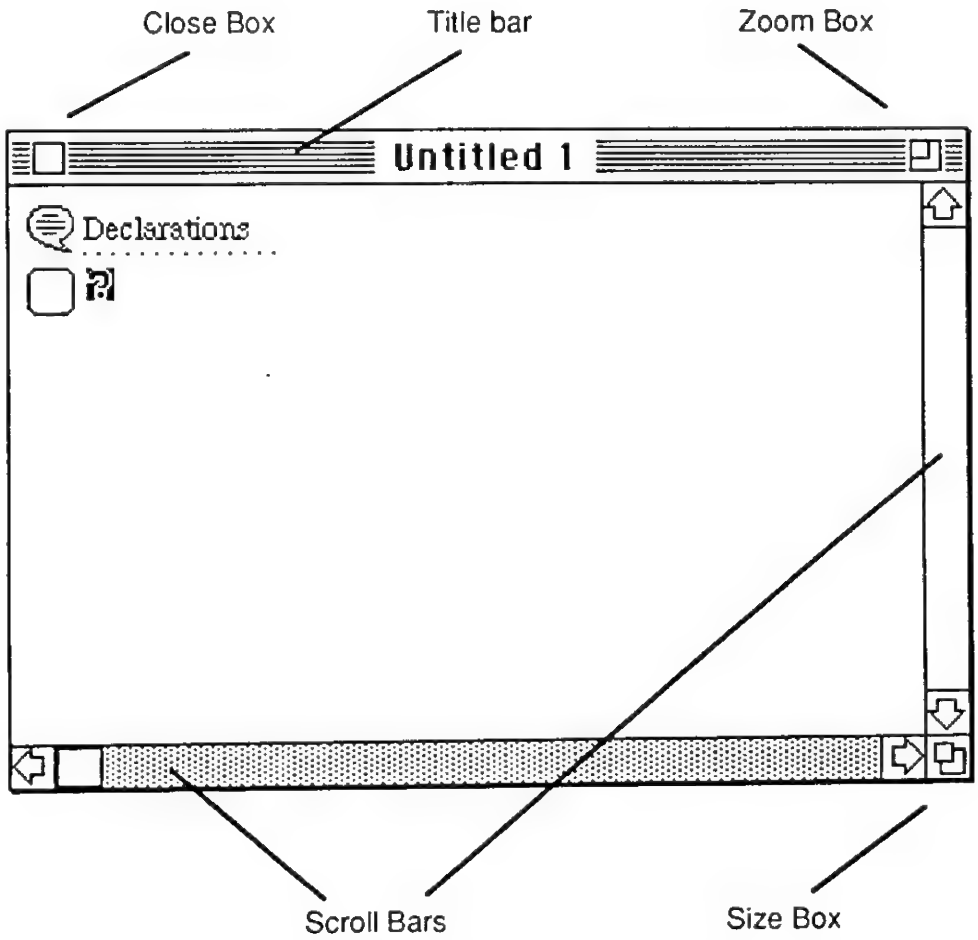
Each notebook is a self-contained theoretical world displayed in a standard Macintosh window that you can move, re-size, and print.

This chapter describes:

- Notebook files
- Notebook management
- The notebooks included on your Theorist distribution disks

Notebook Files

When you first open Theorist, the default display is a single untitled notebook.



A notebook file is displayed in a standard Macintosh window with scroll bars and arrows, a zoom box, size box, and close box. Several notebooks can be open at once; however, only the front-most window is active. To move a notebook window, click on the title bar, drag the window to a new location, and release.

The default "Untitled 1" notebook contains a comment proposition titled "Declarations." The dotted line under the title indicates that daughter propositions are currently hidden. To show (or hide) these propositions, double-click on the proposition icon. The default notebook contains eleven name declarations. For more information about propositions and name declarations, see the Propositions chapter.

Use the commands on the File menu to create new notebooks and open, close, save, and print notebook files as you would word-processing files.

Startup Notebooks

You can customize your Theorist work environment in several ways. If you create one or more notebooks with the titles "Notebook 0", "Notebook 1", "Notebook 2", and so on, up to "Notebook 10", and keep these files in the same folder as the Theorist application, every time you launch the program directly (by double clicking on the application icon), these notebooks also open. When you quit Theorist, any such notebooks are automatically saved. (All other notebooks present a dialog box, asking if you want to save your changes.)

Note: If you launch Theorist by double-clicking on a notebook file (rather than double clicking on the application icon), startup notebook files do not open (even if that notebook is a startup notebook). Startup notebooks only open if you launch the program from the application icon.

If you do not want to save your changes to one of these notebooks, choose **Revert** from the **File** menu before closing the notebook. Otherwise, your changes are saved automatically.

By automatically opening (and saving when quitting), startup notebook files let you easily maintain several streams of work.

Stationery Files

Using stationery files is another powerful way to customize your work environment. If you find that you regularly want to start a new notebook with a particular set of rules and declarations (other than the minimal default set), create a "stationery file" containing these rules and declarations.

To create a stationery file, modify a notebook to include the particular set of declarations and preferences you want, then choose **Save as Stationery** from the **Edit** menu. This creates a special file that acts like

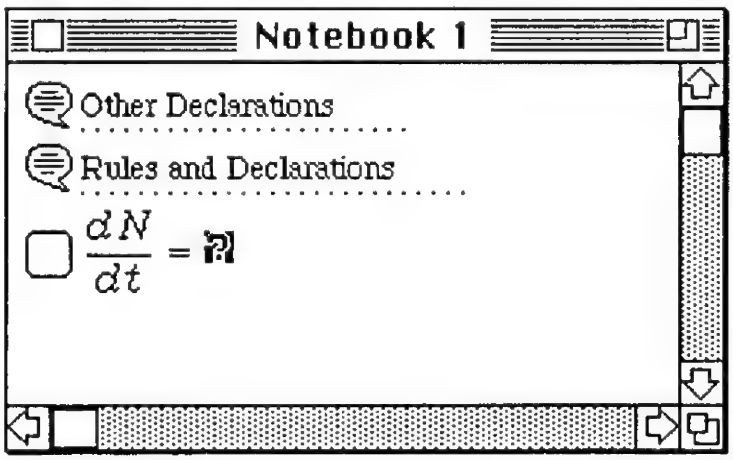
a pad of paper. When you open a stationery file, you effectively "tear off" a new untitled notebook with all the declarations and preferences you created.

If you name a stationery file "Notebook 1" (or "Notebook 2", etc.), and keep this file in the same folder as the program, a new untitled notebook file containing that set of rules and declarations is created every time you launch the program.

Managing Notebooks

You can use the outlining structure in a notebook to keep your notebooks organized. This section describes one structure that several people have found useful. Many other organizations are also possible.

This arrangement consists of three main areas: a comment proposition for temporary or experimental declaration, a second comment proposition for established rules and declarations, and an area for ongoing work.



Other Declarations

Each time you declare a new name, its name declaration is placed at the top of the current notebook. If there is a comment proposition at the top of the notebook, the new name declaration is positioned as a daughter proposition to the comment proposition. Experiment with name declarations, editing and revising them until they work just the way you want. Then you can organize them under the Rules and Declarations comment.

Rules & Declarations

The second comment proposition in this scheme is used to contain an organized structure of names and rules. Some of these might be copied from the notebooks included with Theorist on your distribution disks. And some will be names and rules you develop on your own.

Ongoing Work

After the two comment propositions is an area for ongoing research. In this space you create assumptions and conclusions, manipulate expressions, and create graph theories and animations. Use this space as you would an infinite blackboard or pad of paper.

If you do not want to save the results of a particular investigation, select one or more propositions and press Delete. Deleting a proposition with daughter propositions deletes those dependent propositions as well. You can safely press Delete repeatedly. Any collapsed proposition (hiding its daughter propositions, and shown with a dotted line underneath) must be explicitly selected to be deleted. the Comment proposition(s) at the top of a notebook are usually collapsed and therefore safe from unintended deletion.

Distribution Notebooks

Several notebooks are included on your Theorist distribution disks. These notebooks contain examples of colorful Surface plots as well as a wide range of mathematical rules and functions.

These notebooks are organized by hierarchical "bundles." A comment proposition acts as a title and handle for a group of daughter propositions that contain useful rules and declarations. To move a bundle from these distribution notebooks, select the comment proposition of the bundle that you want, Copy it, and Paste it into your research notebook.

If there are any name conflicts (e.g., a bundle contains a previously declared name), a message announces that there is a conflict. The first instance of the conflicting name is selected. If the name declarations are identical (e.g., the bundle and your notebook both declare sin as the standard sine function), delete the selected name.

Only the first naming conflict is announced. More conflicts may exist. Usually this is not a problem. The next time a manipulation requires the conflicted name, an appropriate warning is displayed asking that one of the declarations be deleted. You can also flush out any possible naming conflicts by clarifying the notebook repeatedly until all conflicts are reported and cleared.

If a name is used in two different ways (e.g., you have created a function named G and the bundle declares G to be a different function), delete the newly pasted bundle, change the conflicting name declarations in your notebook, and re-paste the bundle. To change the spelling of a name, open the Declarations comment at the top of the current notebook and

find the name declaration proposition for that name. Then change the spelling in that proposition. Changing the spelling or behavior of a name in its declaration is immediately reflected throughout the notebook.

"Standard Rules and Declarations," in the "Mathematics Notebooks" folder, is a collection of rules and declarations you might find useful. It contains a subset of rules from a variety of distribution notebooks, including "Trig Functions," "Log and Power Functions," and "Hyperbolic Functions."

For more information about name declarations and Theorist's outlining tools, see the Propositions chapter.

See Appendix D for a complete list of notebooks included on the Theorist distribution disks.

Propositions

Notebooks contain a set of propositions organized hierarchically. Propositions can contain other propositions and are often related to other propositions in various ways. Each notebook always contains at least one proposition, a Main Theory proposition. The main theory is essentially equivalent to a notebook. All propositions that you enter are contained within the main theory.

All types of propositions are presented in this chapter:

- Statements
- Working Statements
- Case Theories
- Graphs and Plots
- Comments
- Name Declarations
- Independence Declarations
- Transformation Rules

The chapter ends with a discussion of the tools you can use to organize and arrange your propositions.

To find out more about any proposition, select it and choose **Get Info** from the **Notebook** menu (or press Command-I). If the proposition is derived from other propositions, you can move to those propositions through a sort of hypertext network. The hierarchical display of propositions in a notebook presents the overt story of mathematical investigation and discovery. Using Get Info, you can trace the hidden connections and logical dependencies between propositions.

Statements ▢ △

A statement is an expression in a Theorist notebook. Every statement is either an "assumption" or a "conclusion." Every statement you enter is an assumption. Conclusions are logically derived from one or more statements. To create a new assumption, press Return.

Statements can be expressions (e.g., $3x$, $\log(y)$) or equations (e.g., $x = \sin(y)$, $y = x^3 + 2x + 4$). The following recursive definition defines statements exhaustively. A statement is either:

- A number (e.g., 4, 27, 2.3)
- A name (e.g., x, y, sin)
- A wildcard variable (e.g., $\mathfrak{x}$, $\mathfrak{y}$)
- An op that encloses one or more expressions (e.g., $4 + y$, $x = y$)

Numbers, names, and wildcard variables are the elementary particles of expressions and equations. Ops are used to put these elements together. For more information about ops, see the Expressions chapter.

Assumptions are displayed with a square icon; conclusions with a triangular icon:

▢ $y = \sin(x)$
△ $x = \arcsin(y)$

Each conclusion is the result of a valid manipulation. Only you can create assumptions, and only Theorist can create conclusions. You cannot edit a conclusion; if you try to, the statement is duplicated as a new assumption that you can edit.

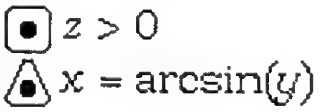
When you manipulate a selected expression, the conclusion is displayed directly beneath that expression, indented to the right. Other statements may be used to derive the conclusion.

You can always discover which statements a conclusion is derived from. Select the conclusion and choose **Get Info** from the **Notebook** menu (or press Command-I). The displayed dialog box states the type of the proposition. Click on the **More Info** button to see a list of all statements used to generate the conclusion.

If you rearrange the statements in a notebook, you do not change their logical relationship. However, if you delete a conclusion, all of its dependents turn into assumptions. Also, if you modify an assumption, all conclusions directly derived from it become assumptions.

Working Statements

Working statements are assumptions or conclusions used by a manipulation to generate a new conclusion or used to generate a graph. You can specify which statements to use as working statements, or they can be designated automatically when you perform a manipulation. Working assumptions and conclusions are displayed with square and triangular icons, respectively. They can be distinguished from ordinary statements by the dot contained within the icon:



All working statements provide specific information about the name on the left-hand side of the statement.

To make a statement a working statement, select it and choose **Make Working Stmt** from the **Notebook** menu (or press Command-D).

In some instances, an assumption or conclusion is automatically promoted to a working statement. For example, if you are calculating the value of an expression that contains the name *y*, Theorist searches the current theory (and all enclosing theories) for a statement of the form *y* = *value*. The first such statement becomes a working statement and its icon is marked with a dot.

The search for a possible working statement proceeds through the current theory from the top to the bottom, then the enclosing theory (if any) from top to bottom, and so on.

You can change a working statement into an ordinary statement by selecting it and choosing **Stop Working Stmt** from the **Notebook** menu. You can also turn a working statement off by selecting an alternate statement of the same type that defines the same name, and making the new statement a working statement. There can be only one working statement of each type for a particular name in a particular theory.

There are six types of working statements:

Statement Type	Example
Value	$\boxdot x = 5$
Maximum	$\boxdot x < 5$ $\boxdot x \leq 10$
Minimum	$\boxdot x > 5$ $\boxdot x \geq 10$
Increment	$\boxdot dy = 0.0001$
Function	$\boxdot f(\Sigma 3) = \sin(\Sigma 3)$
Order	$\boxdot y^5 = 0$

Value Statement
$\boxdot x = 5$

The Value Statement is the most common working statement. It defines a working value for a name. The name must be of class Constant, Variable, or M-Linear Operator. (It cannot be a D-Linear Operator or a Function.)

Value statements are used by the Calculate manipulation and to create graphs. Symbolic manipulations (e.g., Simplify, Factor) do not evaluate expressions and therefore do not take into account value statements.

Maximum Statement

◉ $x < 5$
◉ $x \leq 10$

Minimum Statement

◉ $x > 5$
◉ $x \geq 10$

Maximum and Minimum statements define an exclusive or inclusive upper or lower bound for a name. The name must be of class Constant or Variable. Only one maximum statement and one minimum statement can be in force at a given time for a given name.

Maximum and minimum statements are often used to control the original display of graphs. If you plot a function, such as $y = \sin(x)$ in two dimensions, the graph theory usually uses default bounds of $x > -3$ and $x < 3$. Once created, graphs can be easily altered, but you can also set larger or smaller bounds explicitly before creating a graph.

If you only create a single maximum or minimum statement (e.g., $x > 0$), graphs are created using this as one of the boundary values; the other is derived from internal heuristics.

If Auto Casing is on, maximum and minimum statements can also affect the Move Over and Isolate manipulations. If alternate solution values for these manipulations are within the range determined by a maximum and minimum statement, more than one case theory is generated. For example, if $x > 0$ is a working statement, $x^2 = 9$ has only one solution: $x = 3$. Otherwise, two case theories are generated for $x = 3$ and $x = -3$.

The iterative ops, Summations (Σ) and Pi Products (Π), are also affected by minimum and maximum statements. For example, if $k \leq 3$ is a working statement, and you expand the summation of x^k as k goes from 1 to 10, the solution is as follows:

$$\sum_{k=1}^{10} x^k = x^3 + x^2 + x$$

Increment Statement

☐ $dy = 0.1$

The Increment Statement defines an increment value for a variable. The name must be of class Variable. The increment value is used for numerical calculations of derivatives. In the absence of an increment statement, a default value of 0.0001 is used.

If the increment value for a variable is set to zero (e.g., $dy = 0$), the variable (y) acts as a constant and all derivatives taken with respect to it are zero in that theory.

Function Statement

$\square f(x) = x^2$

The Function Statement defines a functional relationship between one or more input values and one or more output values. The function name must be of class Function. Function arguments must be wildcard variables (e.g., x, y). (See the Expressions chapter for more information.)

Function statements are used to create graphs. They are also used by the Calculate manipulation to numerically evaluate the function. Symbolic manipulations do not take function statements into account.

If a function statement has more than one argument, all occurrences of that function must have the same number of arguments. You can also pass functions a vector as a single argument that contains more than one element. In the following examples, x and y are single arguments, whereas p is a two-element vector:

$$\bullet f(x,y) = \left(\sqrt{x^2+y^2}, \arctan\left[\frac{y}{x}\right]\right)$$

$$\bullet f(p) = \left(\sqrt{p_1{}^2+p_2{}^2}, \arctan\left[\frac{p_2}{p_1}\right]\right)$$

Usually it is preferable to create function statements with vector Wildcard variables so that you can pass in vector expressions such as the product of two matrices. Use an equation of the second form to pass vector arguments.

Order Statement

$\bullet y^5 = 0$

The Order Statement reduces to zero the powers of a name greater than a designated value. The name must be of class Constant, Variable, or M-Linear Operator. Use order statements only if the absolute value of the name (or, if the name is a matrix or M-Linear Operator, the largest eigenvalue) is much smaller than one.

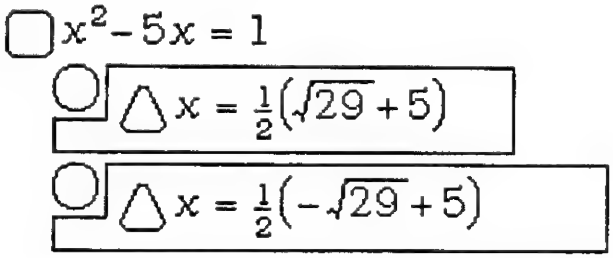
The order statement $y^5 = 0$ declares that y^5 and all greater powers of y can be reduced to zero. This statement is taken into account by the Simplify and Expand manipulations.

Case Theories

A theory is a list of related propositions. Every notebook consists of a Main Theory that contains propositions, including all "case theory" propositions. Case theories can be nested in any hierarchical fashion and are displayed with a circle icon and a variable sized box containing all internal propositions.

Case theories are self-contained work areas that you can use to explore divergent trains of thought. For example, in a particular mathematical situation, you can explore a world where $x = 3$ and another where $x = 2$.

If Auto Casing is on (choose **Auto Casing** from the **Prefs** menu), any manipulation that generates multiple solutions creates a case theory for each solution. For example, if you solve the following equation for x, both solutions are displayed:



Generating Case Theories

Only two manipulations, Isolate and Move Over, generate case theories, and only when Auto Casing is on. If Auto Casing is off, only the first solution is given within the current theory.

Manipulations that would generate a large number of case theories may display a single solution that contains arbitrary constants. For example, the solution in x of $x^5 = 1$ is either $x = 1$ (if Auto Casing is off) or (if Auto Casing is on):

$$x = e^{2/5 n_{100} \pi i}$$

The second solution includes the arbitrary constant n_{100}, which indicates an arbitrary integer value. For more information on arbitrary constants, see the Expressions chapter.

Creating Case Theories

You can also create any number of case theories by choosing **Case Theory** from the **Input** menu. All selected statements (and conclusions based on those statements) are included in the new case theory. If no statements are selected, a single empty assumption appears in the new case theory:

Because Theorist is non-procedural, you can enter conflicting assumptions into a single theory, such as $x = 1$ and $x = 0$. In practice, it is best to put conflicting assumptions in alternate case theories. If you perform a manipulation in a theory that contains conflicting assumptions, you can end up with confusing and contradictory results.

In a few instances, it is useful to have conflicting statements within a single theory. If you approximate the value of an expression (e.g., by creating a Taylor series), this appears as a strict equality:

$$\square\ y = \sin(x)$$

$$\triangle\ y = -\frac{1}{5040}x^7 + \frac{1}{120}x^5 - \frac{1}{6}x^3 + x$$

Seventh Order Taylor Series

Using the Substitute and Move Over manipulations, you can determine the closeness of the approximation for various values of x ($x = 0.1, 1, 10$):

$\square\ y = \sin(x)$ — Taylor Series

$\triangle\ y = -\frac{1}{5040}x^7 + \frac{1}{120}x^5 - \frac{1}{6}x^3 + x$ — Substitute

$\triangle\ -\frac{1}{5040}x^7 + \frac{1}{120}x^5 - \frac{1}{6}x^3 + x = \sin(x)$ — Move Over

$\triangle\ -\sin(x) - \frac{1}{5040}x^7 + \frac{1}{120}x^5 - \frac{1}{6}x^3 + x = 0$

$\square\ x = 0.1$ — Substitute

$\triangle\ -\sin(0.1) + 0.099833 = 0$ — Calculate

$\triangle\ -2.7555 \times 10^{-15} = 0$

$\square\ x = 1$ — Substitute

$\triangle\ -\sin(1) + \frac{4241}{5040} = 0$ — Calculate

$\triangle\ -2.7308 \times 10^{-6} = 0$

$\square\ x = 10$ — Substitute

$\triangle\ -\sin(10) - \frac{82370}{63} = 0$ — Calculate

$\triangle\ -1306.9 = 0$

Usually you specifically indicate the equations and definitions used for a manipulation. However, if a particular proposition is needed for a manipulation, and cannot be found in the current case theory, the manipulation may search containing case theories for the needed definition. Case theories are searched from the innermost to the outermost, and from top to bottom within each theory. If an external statement is used as a working statement, it's icon is marked with a dot.

The following propositions are available to any contained case theory:

- Working statements
- Name declarations
- Independence declarations
- Transformation rules

Case theories always describe more specific conditions than their enclosing theory. Most case theories start with one or more equations that distinguish the theory from its rival theories.

Graph theories are also theories. They contain propositions describing the various graphical elements that make up a graph. Most of these propositions are only valid inside graph theories. You can also create case theories within graph theories. See the notebook "Spherical Family" in the Graphics folder on the Theorist distribution disks for an example of this.

Graph Theories

Two- and three-dimensional graph theories are special theories, and all plots are propositions contained within graph theories.

Graph theories, like case theories, are displayed with a variable box containing all internal propositions. The icons for graphs and plots are:

Line plot

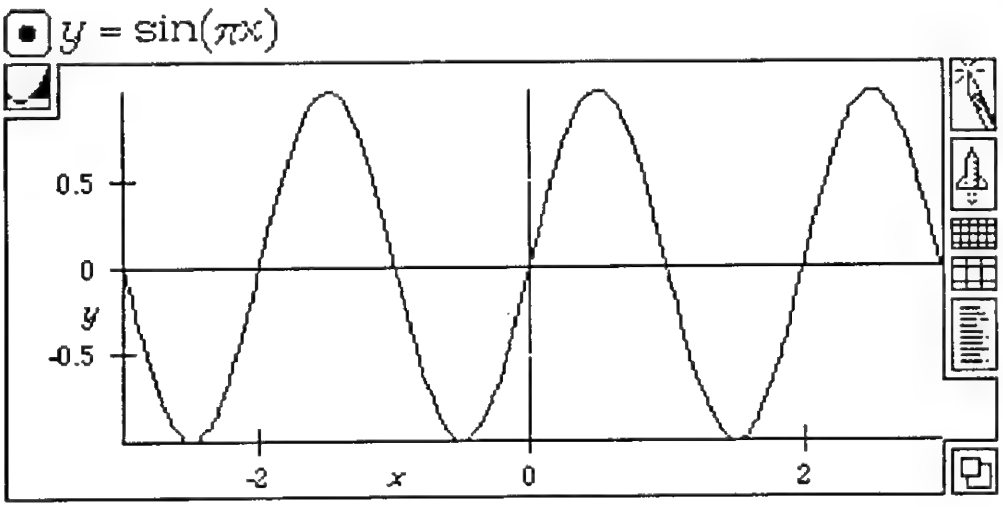
Contour plot

Surface plot

Two-Dimensional graph theory

Three-Dimensional graph theory

The following is a simple two-dimensional graph theory displaying a single plot:



For more information on graph theories and plot propositions, see the Graphs chapter.

Comments

Comment propositions contain text or graphics used to annotate your notebooks. They are ignored during all manipulations and operations.

Comment propositions are displayed with a balloon icon:

Text comment proposition

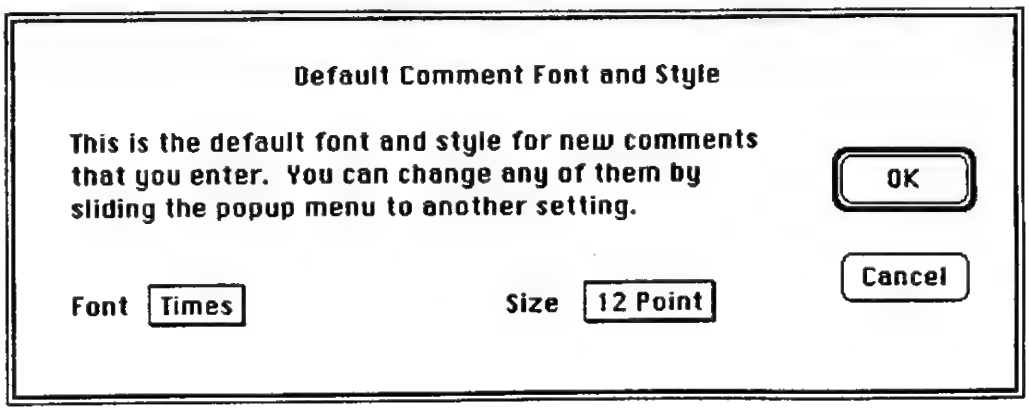
Picture comment proposition

Text Comments

To create a text comment proposition, press Shift-Return. Pressing Return alone always creates a new assumption proposition. (If you create a new assumption by accident, immediately press Enter and it turns into a text comment.)

To alter the font, size, style, and color of the text choose the appropriate commands from the Edit menu. To enter a hard carriage return within a text comment (i.e., to force a new line), press Shift-Return again.

You can also specify the font and size of all future text comment propositions. Choose the **Comment Default** option on the **Prefs** menu and the following dialog box comes up:



To adjust the size and shape of a text comment, click down in the lower right-hand corner of the comment and drag as needed. If you cannot

find the invisible size box, click on the icon; the size box is at the lower right-hand corner of the highlighted area.

The following is a text comment proposition:

The following graphs display a family of spherical harmonics with constant amplitude and increasing frequency.

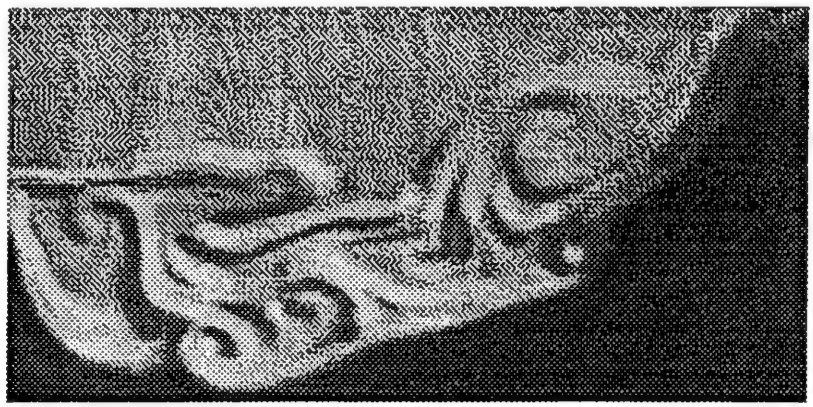
The first proposition in a new notebook is the comment proposition called Declarations. It contains a set of name declarations as daughter propositions. Each new name that you declare is placed within this proposition. If you delete this comment proposition, new names are placed at the top of the notebook.

Picture Comments

Picture comment propositions can contain any graphic stored on the Macintosh clipboard. To create a picture comment, Paste the graphic contents of the clipboard into a notebook. Unlike a text comment, you cannot create an empty picture comment proposition.

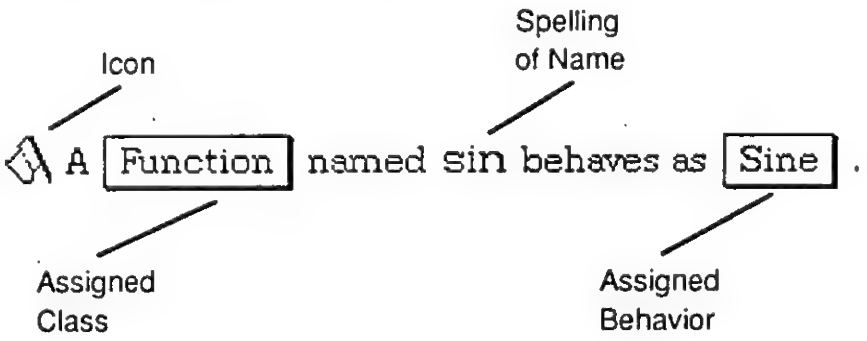
To take a snapshot of a graph, select it and choose **Copy as PICT** from the **Edit** menu. When you Paste it into a notebook, it is displayed as a picture comment proposition.

You can store any PICT graphic from any source as a picture comment. For example, the following was taken from a supercomputer simulation of fluid dynamics:



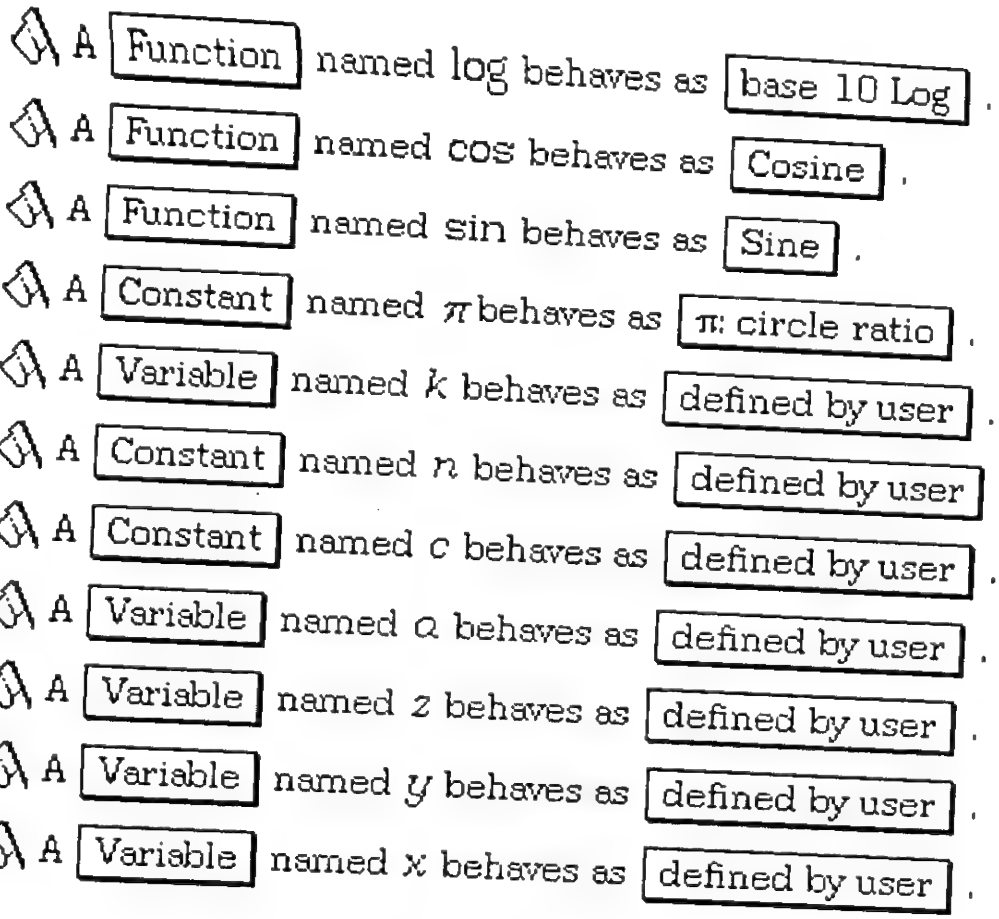
Name Declarations

Name declaration propositions designate the class and behavior of each name in a theory. Names can consist of Roman characters, Greek characters, and the symbols ∞, ⟨, ⟩, ℏ, ℜ, ℑ, ℵ, and ′. Each name declaration proposition appears with a flag icon:



This proposition declares that the name "sin" is of class Function and has the predefined behavior called Sine. Names can be of class Constant, Variable, M-Linear Operator, D-Linear Operator, or Function. The behavior of a name can be one of fifty-seven predefined behaviors or as you define. Defining the behavior of a name can be as simple as creating an equation using that name. The statement $x = 13$ defines the behavior of x.

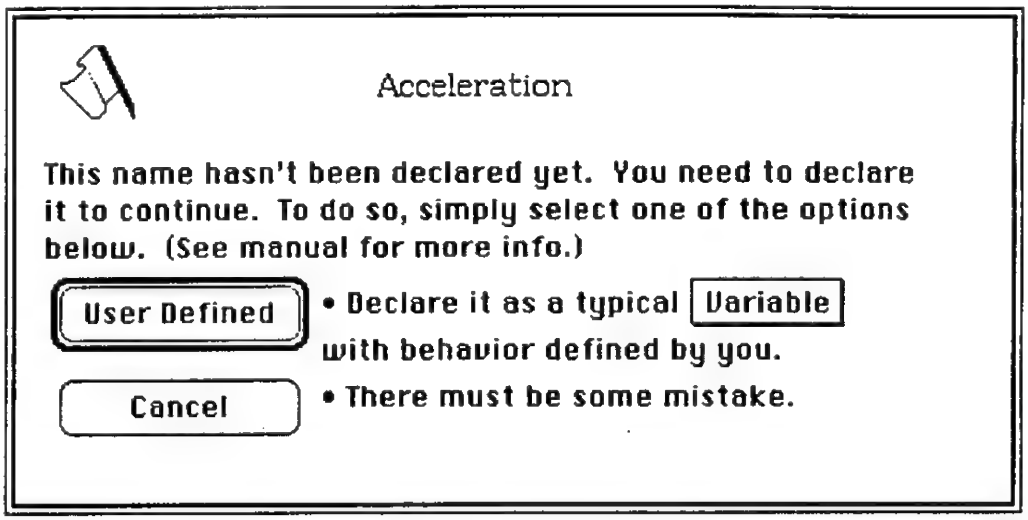
Each notebook initially contains eleven name declarations:



All name declarations are contained in the "Declarations" comment proposition, shown at the top of each notebook. (If you move or delete this comment proposition, new name declarations are placed at the top of the current notebook.)

To declare the names in a notebook, select **Clarify** from the **Notebook** menu (or press Command-Space). A dialog box appears in which you can create name declarations for all undeclared names. If a name has not been declared when it is first involved in a manipulation (or used to create a graph), this dialog box appears asking you to declare the name.

For example, if you use the name Acceleration in a notebook, the first time you manipulate an expression (or Clarify the notebook) the following dialog appears:



Accept the default (click User Defined or press Return) to declare Acceleration as a variable. Use the pop-up menu on the right of the dialog box to indicate the name's class (Constant, Variable, M-Linear, D-Linear, Function).

To edit a name declaration, open the Declarations comment (double-click on its icon). You can change the class or behavior by clicking on the pop-up menus in the name declaration proposition. You can also change the spelling of a name. Changing the spelling, class, or behavior of a name is immediately reflected throughout a notebook. However, changing the class or behavior of a name may invalidate existing derivations; manipulations are *not* re-executed.

There are fifty-seven predefined names with fifty-seven associated predefined behaviors. To use a predefined name, enter the name and accept the default behavior associated with it when the dialog box appears. To assign a predefined behavior to a different name, enter the predefined name associated with the behavior that you want, accept the default associated behavior, then edit the name's spelling. For more information on predefined names and behaviors, see the Expressions chapter.

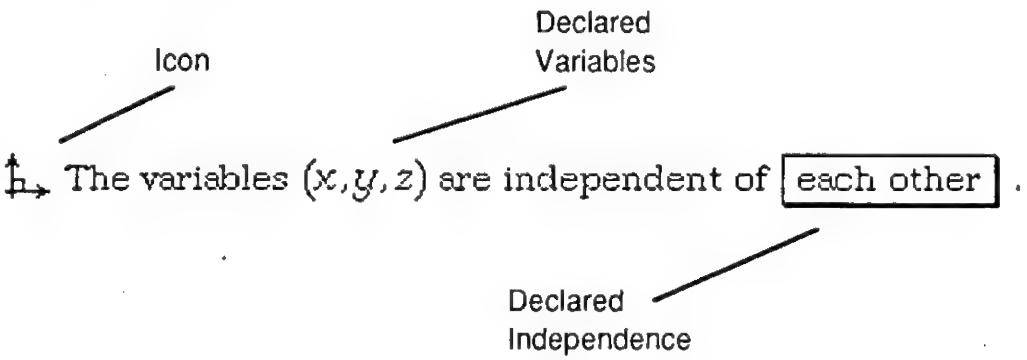
Wildcard variables are not names and should not be used in name declarations.

Independence Declarations

Independence Declaration propositions assert that certain variables are independent of each other (or are independent of all variables). If two variables are independent of each other, the derivative of one with respect to the other is zero; there is no relationship linking them together. By default, all variables in Theorist are dependent on one another.

Independence declarations can also declare that two M- or D-Linear Operators commute with each other. Declarations of this type are only used by the Commute manipulation, not Simplify.

Independence declarations are shown with an icon displaying two orthogonal axes:



This proposition asserts that *x*, *y*, and *z* are independent of each other. Their derivatives with respect to each other are zero.

You should declare variables independent of each other for any situation that has multiple variables whose derivatives with respect to each other should be zero, such as:

- Spaces with multiple orthogonal dimensions
- Multiple integrations such as: $\int_0^1 \left(\int_0^1 x\,y\,dx \right) dy$
- Two-dimensional (or higher dimension) Taylor series

The pop-up menu at the end of the proposition can declare the given variables to be independent of each other (the default) or independent of all variables. If you use the Summation (Σ) and Pi Product (Π) ops, you may want to declare the iteration variable (usually *k*) to be independent of all variables, especially if you take derivatives of expressions containing the iteration variable.

For example, you should probably declare *k* independent of all variables for simplifying the following expression:

$$\square \frac{\partial}{\partial x} \sum_{k=1}^{\infty} x^k$$

Without an independence declaration, the Simplify manipulation produces the following:

$$\triangle \frac{\partial}{\partial x} \sum_{k=1}^{\infty} x^k = \sum_{k=1}^{\infty} \left(x^k \ln[x] \frac{\partial}{\partial x} k + k x^{k-1} \right)$$

If you declare *k* independent of all variables:

↳ The variables k are independent of [all variables] .

the result is much more manageable:

$$\triangle \frac{\partial}{\partial x} \sum_{k=1}^{\infty} x^k = \sum_{k=1}^{\infty} k x^{k-1}$$

For more information on the Summation and Pi Product ops, see the Expressions chapter.

Use independence declarations with caution. If x and y are declared independent, you cannot create a graph of the relationship between these two variables, even if you explicitly state that $y = \sin(x)$. The dependent variable (y) is assumed to be a constant with respect to the independent variable (x); the graph produced is a straight line.

Independence declarations take precedence in any theory (or enclosed case theory).

A variable cannot appear in more than one independence declaration (unless the declarations are in sibling case theories). For example, if you have two coordinate systems (x, y, z) and (x', y', z), where (x, y) and (x', y') are rotated around the z axis relative to each other, with z shared between the two coordinate systems, you cannot make two independence declarations both listing z. Instead, create a variable z', make $z = z'$ a working statement, and create two independence declarations:

The variables (x, y, z) are independent of [each other].

The variables (x', y', z') are independent of [each other].

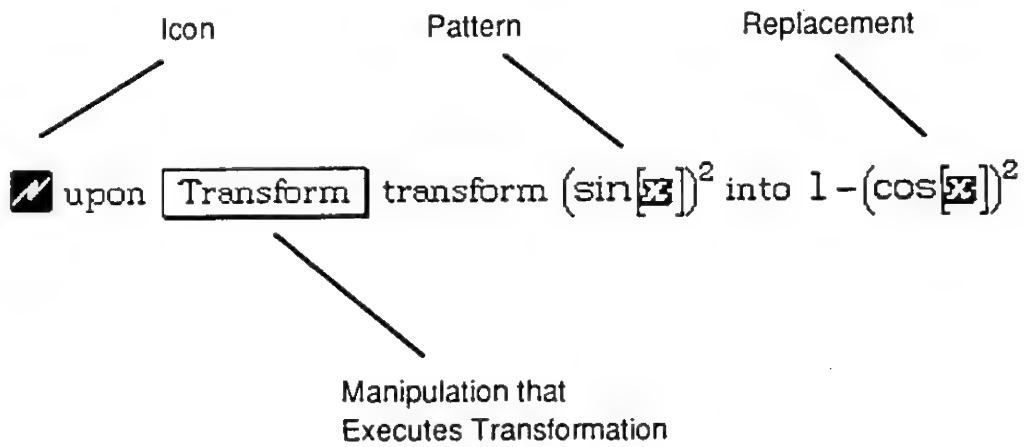
To create an independence declaration, choose **Independence Decl.** from the **Input** menu. If variables are selected when you create the declaration, they are inserted in the declaration.

Transformation Rules

Transformation rule propositions describe transformations invoked when you perform a Transform manipulation. Transformation rules provide a means to extend the capabilities of Theorist. You can designate different transformation rules to execute when the following manipulations are performed:

- Transform
- Simplify
- Expand
- Collect (executes Expand rules in reverse)

Transformation rules are displayed with a lightning bolt icon:



Use the pop-up menu to designate which manipulation invokes a particular transformation.

Each transformation rule describes the transformation of an expression or equation (the pattern) into another expression or equation (the replacement) that is logically equivalent. (Of course, you *can* create fallacious transformation rules. These rules let you create powerful new manipulations or shoot yourself in the foot.)

The pattern can contain regular variables or wildcard variables. Regular variables match the named variable explicitly, whereas

wildcard variables match any variable of a given type, or even entire expressions. (See Wildcard variables in the Expressions chapter.)

Transformation rules are similar to—but more powerful than—the Substitute manipulation. All transformation rules in the current theory (and all enclosing theories) execute when you invoke the Transform manipulation.

To create a transformation rule proposition, choose **Transform Rule** from the **Input** menu (or press Command-Y). If one or more equations are selected, one transformation rule is created from each equation. The left-hand side of an equation becomes the pattern and the right-hand side becomes the replacement. If no equations are selected, a transformation rule is created with two question marks:

Edit existing transformation rules as you would any other proposition.

For more information on transformation rules, see the Manipulations chapter.

Outlining

All propositions in a notebook are arranged in a hierarchy. You can rearrange propositions as you would topics in a word-processing outliner. Each proposition can have sister, daughter, granddaughter, or inline relatives. The following example is created entirely from comment propositions to show the types of outline relationships:

To move a proposition, select it (by clicking on its icon), then hold down the Command key. The mouse pointer becomes a pointing hand. Use this pointing hand to click on the icon of a selected proposition and drag it to a new location. Possible release points highlight. Release the mouse where you want to put the proposition. You can also move selected propositions to the left or right by pressing Command-[and Command-], respectively.

You can rearrange propositions within any case theory by selecting and dragging. However, you must Cut (or Copy) and Paste propositions to move them from one theory to another.

To regroup several propositions, shift-click to select more than one proposition, click on any of the selected propositions with the pointing

hand, and move the group to a new location. The propositions are displayed in the order you selected them.

To collapse the daughters under a proposition, select the proposition and choose **Collapse** from the **Notebook** menu (or press Command-\). A dotted line is displayed beneath the first line of a collapsed proposition. To expose a collapsed family, select the proposition and choose **Expose** from the **Notebook** menu (or press Command-`). You can also double-click on a selected proposition icon with the mouse pointer to collapse or expose a family of propositions.

Collapsed families are called "bundles." The notebooks on your distribution disks contain a number of bundles with useful name declarations. To copy a bundle into your notebook, select it by its title (or main proposition) and use Copy and Paste.

To expose all propositions in a notebook, choose **Expose All** from the **Notebook** menu. Conversely, choose **Collapse All** to collapse all propositions in a notebook.

Expressions

Expressions are Theorist's fundamental objects. This chapter begins with a discussion of the class scheme Theorist uses to manage expressions. The chapter then presents all types of expressions and expression components, including:

- Numbers
- Names (including all predefined names)
- Ops
- Wildcard variables
- Collapsars
- Parentheses
- Undefined values

Class

Theorist can represent a wide variety of objects, ranging from simple numbers to abstract operators for which there is no concrete numerical value. Theorist uses a class system to determine what rules to use when manipulating an object. Each object belongs to one of five classes. This section discusses the five classes:

- Constant
- Variable
- M-Linear Operator
- D-Linear Operator
- Function

The class of an expression determines how it can be used. For example, D-Linear Operators are multiplied onto an argument, whereas functions contain or enclose their arguments, and constants have no arguments at all. See the Summation op (a D-Linear Operator) and Sine (a function) described later in this chapter.

You must declare the class of named objects the first time you manipulate an expression containing a new name or create a graph that uses the name. You can also declare any undeclared names deliberately by choosing the **Clarify** option on the **Notebook** menu. Each object is manipulated according to the rules of its declared class. The default class of a new object is Variable. However, you may get erroneous results if this is not the class you intend.

For example, if you create two matrices *A* and *B* and accept the default class Variable (rather than declaring them to be M-Linear Operators), the expressions *AB* and *BA* are equivalent. But they are not necessarily equivalent because matrix multiplication is not commutative. To avoid errors of this type, you must assign named objects to the correct class.

To see the class of any expression, select the expression, then choose **Get Info** from the **Notebook** menu (or press Command-I).

The class of a particular expression depends on what the expression is, and the classes of any subexpressions it contains. For example, if a

Variable x is added to a matrix, an M-Linear Operator, the result is an M-Linear Operator. However, if the matrix contains one or more D-Linear Operators, the object as a whole is a D-Linear Operator.

Each predefined name has a preset class. For example, π is always of class Constant and arctan is always of class Function.

Constants

A constant is a scalar (a single value) whose derivative is zero. Constants do not need to be defined to have a specific value. You can treat them as abstract unknowns.

If you create a working definition for a constant named c, it has the value you assign to it. Its actual value may not be constant (it may depend upon other variables) but it is treated as a constant for manipulations and for creating graphs.

All numeric values are constants as well as certain predefined names (e.g., π, e, ∞, i). All expressions consisting entirely of constants (including functions of constants) are also constants.

The following are all of class Constant:

$$5 \qquad \pi \qquad \sqrt{5} \qquad \infty \qquad \arctan(\pi^2)$$

Variables

A variable can take on the value of any integer or real or complex number. A variable's derivative is not necessarily zero. Variable multiplication is commutative. The value of a variable cannot be a matrix or derivative.

The relationship between any two variables is, by default, undefined. However, you can define either in terms of the other with an equation or a working statement, or you can declare them to be independent with an independence declaration.

All names declared as variables are of class Variable, as are all expressions made up of constants and variables.

The following are all (usually) of class Variable:

$$x \qquad y \qquad t \qquad \sqrt{xy}$$

M-Linear Operators

An M-Linear Operator is a matrix or behaves like a matrix in certain ways (e.g., Hermetian operators). Matrix multiplication is associative but not commutative. However, multiplication of a matrix and a variable (or a constant) is commutative.

As an example of non-commutative algebra, if A and B are declared to be M-Linear Operators, and you Expand $(A + B)^3$ the result is:

$$A^3 + BA^2 + B^2A + ABA + B^3 + AB^2 + A^2B + BAB$$

Because multiplication is not commutative for M-Linear Operators, the terms do not recombine when simplified.

Note: If you declare the relative independence of A and B (with an independence declaration) you can reduce an expression of this form. However, you must Commute each expression by hand. Then choose **Simplify** from the **Manipulate** menu to reduce the expression.

All matrices with constant, variable, and M-Linear elements are M-Linear Operators. A matrix with one or more D-Linear Operators as an element is a D-Linear Operator.

The following are all of class M-Linear Operator:

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \qquad \begin{pmatrix} x & y \\ z & 0 \end{pmatrix}\begin{pmatrix} a & b & c \\ d & e & f \end{pmatrix} \qquad \begin{pmatrix} 0 & y \\ -y & 0 \end{pmatrix}^2 + 1$$

D-Linear Operators

A D-Linear Operator is a derivative operator, or an operator that behaves like one. Multiplication of a D-Linear Operator and any other object (other than a constant) is not necessarily associative nor commutative.

D-Linear Operators are multiplied onto the expression immediately preceding them. The three most common D-Linear Operators are the predefined name d, the partial derivative op $(\partial/\partial x)$ and the summation op (Σ). These three all operate by varying the value of particular variables in the expression immediately following them.

D-Linear Operators do not commute with variables, although they do commute with constants. For example:

$$\sum_{k=0}^{10} 5x^k = 5\sum_{k=0}^{10} x^k$$

However:

$$\sum_{k=0}^{10} kx^k \neq k\sum_{k=0}^{10} x^k$$

If the right-most factor in a product is a D-Linear Operator, then the expression as a whole is of that class.

The following are all of class D-Linear Operator:

$$x+\frac{\partial}{\partial x} \qquad \frac{\partial}{\partial x}+x \qquad y\frac{\partial}{\partial x}$$

However, the following are all of the same class as y:

$$\frac{\partial}{\partial x}y \qquad \left(x+\frac{\partial}{\partial x}\right)y$$

If you multiply a D-Linear Operator onto a constant, variable, or matrix, then the class of the new expression is of class Constant, Variable, or M-Linear Operator. That is, if the D-Linear Operator is on the left-hand side of a product, the term as a whole is of the same class as the partial term on the right.

Functions

Functions enclose one or more expressions as arguments. (Actually, functions are just names of class Function. The function op, symbolized by a pair of parentheses, encloses the arguments.) Each function maps an input argument to an output argument. Theorist provides many predefined functions. These are listed later in this chapter in the section Predefined Names.

To create a function of your own design, write an equation that defines its value with wildcard variables as arguments. Then choose **Make Working Stmt** from the **Notebook** menu (or press Command-D).

For example, you can define the function $f(x)$ to be:

$$f(x) = \frac{2^x + 2^{-x}}{e}$$

You can also create functions with more than one argument:

$$f(x, y) = \frac{\sin(x) + \sin(y)}{2}$$

You can also pass a function a vector of elements as a single variable. In the following example, p is a two-element vector:

$$g(p) = \frac{\sqrt{p_1} + \sqrt{p_2}}{2}$$

See the "Special Functions" folder in the Mathematics folder on your Theorist distribution disks for examples of functions and their definitions.

Numbers

Theorist can display and manipulate numeric values in decimal or scientific notation. Type a period for a decimal point, and "e" or "E" to enter a number in scientific notation. You can precede the exponent with a plus or minus sign. For example, to enter 1.23×10^{8}, type "`1.23E8`", or "`1.23E+8`". You can also raise any number to a power using the circumflex (^) character.

Negative numbers are preceded by a minus sign that encloses the number. If you select the minus sign, you select both the number and the sign. The negation op is perhaps the simplest op. All other ops work in a similar fashion. Ops enclose one, two, or more expressions, making of those expressions a single object. To change the sign of a number, select the whole number and press Option-minus, or insert the cursor before the number and enter a minus sign.

Complex numbers (e.g., $4 + 3i$) are represented as a symbolic sum in rectangular coordinates. Numeric matrices are represented as matrices with numeric components. To create a matrix, type an opening parenthesis and separate elements with commas, rows with semicolons. Infinity (∞) and negative infinity (-∞) are also supported. Press Option-5 to create an infinity symbol.

The largest number less than infinity that can be represented is about 1.1×10^{4932}. The smallest number greater than zero that can be represented is about 1.9×10^{-4951}. Overflow and underflow yield infinite and zero values, respectively.

An undefined number is displayed as a question mark (?). It is roughly equivalent to "not a number" or NaN values as specified by the IEEE floating point standard. For example, 0/0 = ?.

Theorist stores numeric values with nineteen digits of precision, enough to measure the age of the known universe to within a second, or to measure the radius of the sun to within one angstrom. You can set the displayed precision of numeric values with the **Display Precision** option on the **Prefs** menu; this setting has no effect on the internally maintained precision.

If you copy an expression out as text (with **Copy** from the **Edit** menu), the numbers contain all nineteen digits of precision. If you copy an expression out as a picture (with **Copy as PICT** from the **Edit** menu), you get a bit-mapped picture of the expression as it appears on screen.

Names

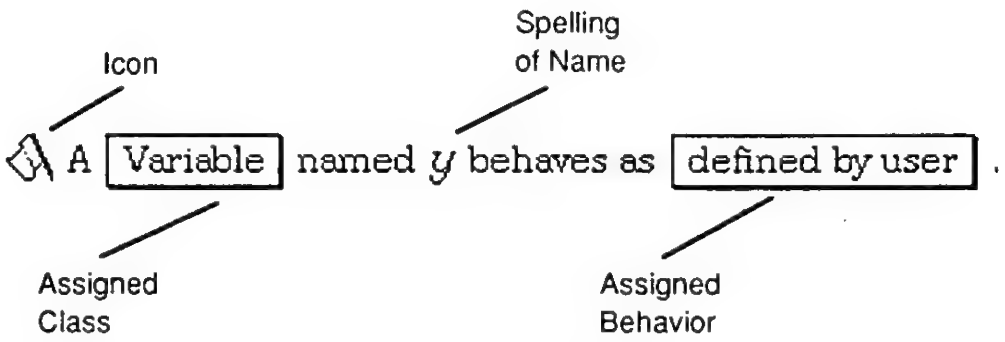
A name is one or more characters that represent a mathematical entity. A name can represent an object of any class (Constant, Variable, M- or D-Linear Operator, or Function).

Simple names can contain one or more of the following:

- Upper- and lowercase Roman characters
- Upper- and lowercase Greek characters
- The symbols ∞ ∇ ⟨ ⟩ ℑ ℜ ℵ ′ ℏ

Names cannot contain digits or any other characters.

The first time you manipulate an expression containing a new name (or create a graph from the expression), a dialog box appears requesting a declaration. Once created, a name declaration appears as a proposition with a flag icon:



A name declaration specifies the spelling of the name, its class, and its behavior. The class must be Constant, Variable, M- or D-Linear, or Function. You define the behavior with a working statement or you can select a predefined behavior.

By default, all new names are placed in the Declarations comment proposition at the top of each notebook. If you do not have a comment proposition at the top of a notebook (named Declarations, or anything else), newly declared names appear at the top of the current notebook.

The behavior of all names is either predefined or defined by you. The predefined names and their associated behaviors are listed later in this section.

Changing a name declaration instantly updates all occurrences of that name throughout a notebook. Names can be spelled any way you want (as long as no two names are spelled the same way). Names are case sensitive (e.g., Log ≠ log).

To edit a name or change its behavior or class, select it and choose **Get Info** from the **Notebook** menu (or press Command-I). A dialog box appears. Click on **More Info** to see the name declaration. If you click on the declaration in the dialog box, the dialog box disappears, the Declarations comment is opened (if necessary), and the selected name declaration is highlighted. Edit the name declaration as you want. Or you can open the Declarations comment directly and search for the name there.

Subscripted Names

Use subscripts to create families of names. For example, instead of using variables *x*, *y*, and *z*, you can use variables x_1, x_2, and x_3. Subscripted names inherit the class of the un-subscripted name, but they can be used in distinct working statements and independence declarations.

Subscripts can be either integers in the range ±32,767 or names. Names must be previously declared and user defined. The class is irrelevant; the name merely serves as a unique identifier. A name can have more than one subscript. A name can have a list (row vector) as a subscript, and a subscripted name can have a second subscript, or a combination. For example, you can use the following sets of variables (which are not equivalent):

- $x_{(1,1,1,1)}$, $x_{(1,1,1,2)}$, $x_{(1,1,1,3)}$ ···
- x_{1_A}, x_{1_B}, x_{1_C}, ···
- $x_{A(1,1,1,1)}$, $x_{A(1,1,1,2)}$, $x_{A(1,1,1,3)}$, ···

Note: The name $A_{(1,2)}$ can refer to a component of a matrix *or* to a unique name. If *A* is a matrix, the value of the subscript (1, 2) determines the matrix element. (See the Index op, below.) If *A* is not a matrix, $A_{(1,2)}$ is a unique name.

If the subscript is a name, and it has a defined value in the range ±32767, Theorist rounds this number to the nearest integer and interprets the subscript numerically.

For example, if x is of class Variable, working statements for x_2 and x_n specify two different variables. If you add another working statement that $n = 2$, the two names become equivalent.

You can also use subscripted names with functions (e.g., $J_0(x)$, $J_1(x)$, $J_2(x)$, ...).

Note: Log and natural log (ln) functions interpret subscripts as the base to use for that function and not as unique names. The base can be any real number or expression. Any name assigned the predefined behavior of the log or natural log functions, uses subscripts in this fashion.

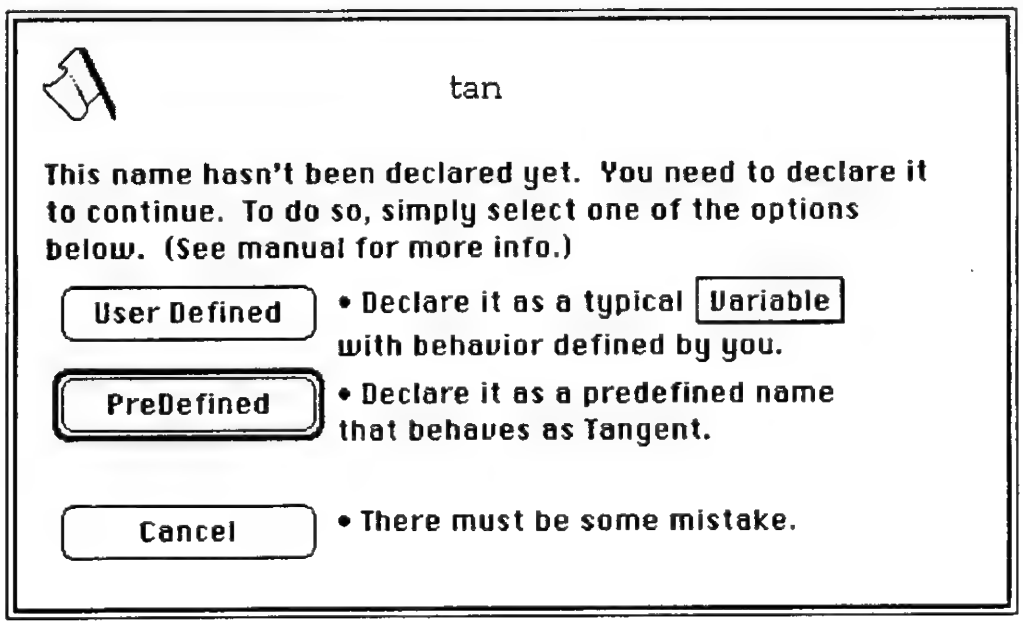
Predefined Names

Theorist provides fifty-seven (57) predefined names. Each name has an associated behavior. Each new notebook contains eleven names. Four of these are associated with predefined behaviors: sin, cos, log, and π, the others are "User Defined."

To use any one of the other predefined names, include it in an expression and when asked to declare it (when you first manipulate the expression or create a graph that uses the expression), accept the default behavior presented in the dialog box.

Predefined names are not "reserved" words. You can associate any name with any predefined behavior, and you can associate a predefined name with a different behavior, either user defined or predefined (although this last is not recommended).

To assign a predefined behavior to a name other than the associated predefined name, proceed as follows. Use the predefined name in an expression and manipulate the expression (or choose **Clarify** from the **Notebook** menu). A dialog box appears, such as:



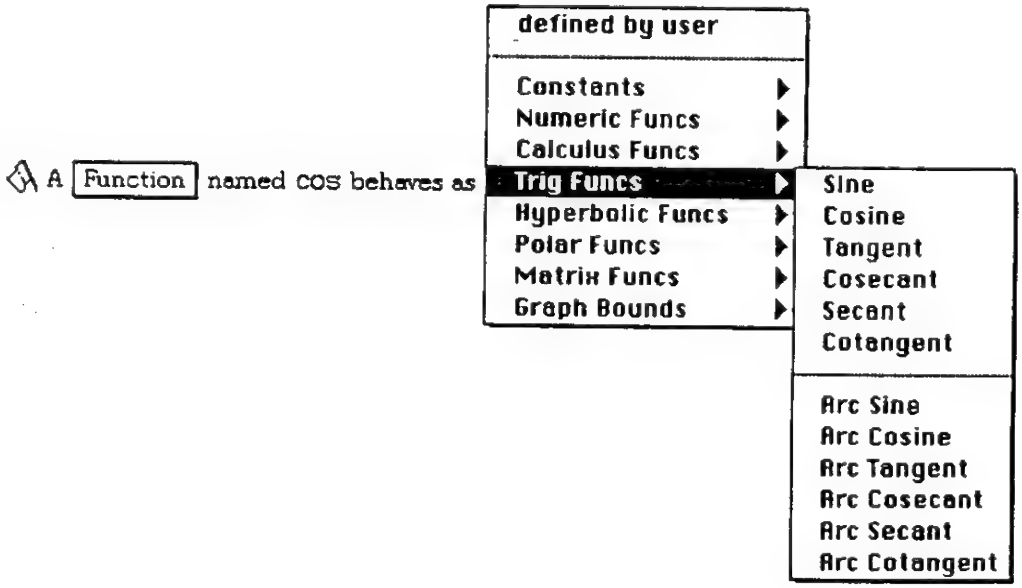
Accept the default, associating the predefined name tan with the predefined behavior Tangent. Open the Declarations comment at the top of the Notebook, and find the newly created name declaration:

A Function named tan behaves as Tangent .

Select the name "tan" and change it to "Tan" or "trigtan" (or whatever) by typing the new name. This change is immediately reflected throughout the notebook.

You can edit existing names in the same way (by opening the Declarations comment), or you can select the name and choose **Get Info** from the **Notebook** menu (or press Command-I) to find the name.

To change the behavior associated with an existing name, adjust the pop-up menu at the end of the name declaration:



The following table lists all fifty-seven (57) predefined names and their associated behaviors.

Predefined Name	Description
Constants	
∞	Infinity
π	Pi: 3.141592653589793238 (actual value used)
i	Square root of negative one ($\sqrt{-1}$)
e	Base of natural logarithms, 2.718281828459045235 (actual value used)
Numeric Functions	
round( )	Rounds argument up or down to nearest integer
floor( )	Rounds argument down to nearest integer
ceiling( )	Rounds argument up to nearest integer
mod( )	Modulus; mod takes two arguments, *x* and *y*, and evaluates to the remainder after dividing *x* by *y*. The result is always between zero (inclusive) and *y* (exclusive).
Re( )	Evaluates to the real part of a complex argument
Im( )	Evaluates to the imaginary part of a complex argument

(Continued)

Predefined Name	Description
Calculus Functions	
log( )	Common logarithm (base 10); defined over (almost) entire complex plane; use subscripts to specify other bases: type "`log_`" and the base
ln( )	Natural logarithm (base *e*); defined over (almost) entire complex plane; use subscripts to specify other bases: type "`ln_`" and the base
exp( )	Value of *e* raised to a given power ($\exp(x) = e^x$); defined over entire complex plane
Γ( )	Gamma function ($\Gamma(x+1) = x!$); defined over entire complex plane (infinite for non-positive integers)
d	Differential operator that takes total derivatives

(Continued)

Predefined Name		Description
Trig and Arc Trig Functions		
sin( )	cos( )	Trig functions that take radian arguments; to use degrees, divide argument by π and multiply by 180; defined over (almost) entire complex plane
tan( )	cot( )	
sec( )	csc( )	
arcsin( )	arccos( )	Anti-Trig functions that return radian results; to get degrees, multiply argument by π and divide by 180; defined over (almost) entire complex plane
arctan( )	arccot( )	
arcsec( )	arccsc( )	
Hyperbolic Functions		
sinh( )	cosh( )	Hyperbolic functions that take real or complex arguments; defined over (almost) entire complex plane
tanh( )	coth( )	
sech( )	csch( )	
arcsinh( )		Anti-Hyperbolic functions that take real or complex arguments; defined over (almost) entire complex plane
arccosh( )		
arctanh( )		
arccsch( )		
arcsech( )		
arccoth( )		

(Continued)

Predefined Name	Description
Polar Functions	
FromPolar( ) ToPolar( )	Functions that convert a 2-vector between polar (r, θ) and rectangular coordinates; r is the radius; θ, the angle, is zero at the positive x axis and increases counter-clockwise
FromCylindrical( ) ToCylindrical( )	Functions that convert a 3-vector between cylindrical (r, θ, z) and rectangular coordinates; r and θ are interpreted as in FromPolar; z is the height and is unchanged by this function.
FromSpherical( ) ToSpherical( )	Functions that convert a 3-vector between spherical (r, θ, φ) and rectangular coordinates; r is the radius; θ starts at the top of a sphere (the "north pole"), increases to $\pi/2$ at the equator, and equals π at the bottom ("south pole"); φ starts at the positive x axis, increases to $\pi/2$ at the positive y axis, equals π at the negative x axis.
Matrix Functions	
RowsOf( )	Returns the number of rows, given a matrix argument
ColumnsOf( )	Returns the number of columns, given a matrix argument
ElementsOf( )	Returns the number of elements, given a matrix argument (RowsOf( ) times ColumnsOf( ))

(Continued)

Predefined Name	Description
Graph Bounds	
left right	Horizontal (first coordinate, usually *x*) minimum and maximum for current 2-D graph, for rectangular coordinates; valid only in 2D graph theories
bottom top	Vertical (last coordinate, usually *y* or *z*) minimum and maximum for current 2-D or 3-D graph, for rectangular coordinates; valid only in graph theories
west east	First coordinate (usually *x*) minimum and maximum for the current 3-D graph, for rectangular coordinates; valid only in 3-D graph theories
south north	Second coordinate (usually *y*) minimum and maximum for the current 3-D graph, for rectangular coordinates; valid only in 3-D graph theories
radius	Maximum radius in all directions; used for polar, cylindrical and spherical plots; valid only in 2-D and 3-D graph theories

Ops

Ops connect expressions in an equation. If expressions are nouns in mathematical sentences, then ops are the verbs. In the expression $a + b$, the Addition op connects the two expressions a and b to create a new expression.

Ops enclose expressions. Unary ops enclose a single expression; binary ops enclose two expressions. Two ops (+ and ×) are *n*-ary: they can enclose any number of expressions (e.g., 1 + 2 + 3 + 4 + 5). When you select an op, you also select the expression or expressions it encloses.

Ops do not perform calculations or manipulations. For example, there is no op that expands a polynomial. To perform this function, select the polynomial, then chose **Expand** from the **Manipulate** menu. Manipulations (such as Expand and Calculate) are actions, whereas ops describe the relationship between names and quantities. Nothing happens spontaneously when you enter an expression or equation; you have to invoke a particular manipulation.

All ops can be entered from the keyboard or the palette. For detailed examples of entering and modifying expressions, see the Editing chapter. In this section, keyboard examples are given for entering the more complex ops.

The following table lists all twenty-three ops. After the table, each op is described individually.

Op	Symbol
Addition	+
Negation	-
Subtraction	-
Multiplication	*
Division	/

(Continued)

Op	Symbol
Equals	=
Relational	$\neq, <, >, \leq, \geq$
Power	x^8
Index (Subscript)	x_8
Absolute Value	$\lvert x \rvert$
Square Root	$\sqrt{x}$
Adjoint	$x^\dagger$
Matrix (Vector)	$\begin{pmatrix} 1 & 2 \\ 3 & x \end{pmatrix}$
Dot Product	•
Cross Product	×
Partial Derivative	∂
Integral	$\int_0^{-\pi} x^2\,dx$
Summation	Σ
Pi Product	Π
Conditional	$\begin{cases} 0 & (x<0) \\ x^2 & (x\geq 0) \end{cases}$
Evaluate At	$\left[\frac{1}{3}x^3\right]_{x=1}^{x=-1}$
Range	–1...1

Addition
$x + 5$

The Addition op (+) can add together any number of terms from 2 to 32,000. Arguments can be any expressions of class Constant, Variable, M- or D-Linear Operator.

You can add a scalar and matrix only if the matrix is square. The scalar is expanded to a diagonal matrix by multiplying it by an identity matrix.

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} + x = \begin{pmatrix} x+1 & 2 & 3 \\ 4 & x+5 & 6 \\ 7 & 8 & x+9 \end{pmatrix}$$

Negation
$-x$

The Negation op (–) is a unary operator that encloses its one operand. Arguments can be any expressions of class Constant, Variable, M- or D-Linear Operator.

Subtraction
$x - y$

The Subtraction op (–) is actually a combination of the addition and negation ops. Arguments can be any expressions of class Constant, Variable, M- or D-Linear Operator.

Multiplication
$5x$

The Multiplication op (*) can multiply together any number of factors from 2 to 32,000. The multiplication op performs different functions, depending upon the class of its operands.

Operand Class	Action
Constant	Ordinary multiplication
Variable	Ordinary multiplication
M-Linear Operator	Matrix multiplication (components must be numeric for calculations); or any abstract operation you define
D-Linear Operator	Derivative or summation; or any abstract operation you define

Multiplication of two M-Linear Operators (e.g., matrices) is not necessarily commutative ($AB \neq BA$). Multiplication of a D-Linear Operator (e.g., d, ∂) and any other operator (other than a constant) is not necessarily commutative or associative: $(AB)C \neq A(BC)$.

To enter $5x - xy$, type "`5*x-x*y`". You can also use spaces in place of the asterisks. In some cases, simple concatenation indicates multiplication. To enter $5x$, type "`5x`". To enter $x5$, type "`x5`".

Division

$\frac{5}{x}$

The Division op (/) is a binary op that can be used with integers, real and complex numbers, and matrices. When you type a slash, Theorist creates a fraction. In the case of matrices (or other M-Linear Operators), A/B is interpreted as $B^{-1}A$. Division by zero almost always yields infinity (∞). Indeterminate forms (e.g., $0/0$, ∞/∞) yield a question mark (?).

Equals

$x = 5$

The Equals op (=) has three uses. Most often, it is used to create an equation, algebraic statement, or working statement. You can also use the Equals op to create relational expressions that can be embedded in other expressions (see below). The Equals op is also used as part of the Evaluate At op (see below).

Expressions can only contain one equals op (unless it is used as part of a relational expression or Evaluate At op).

Note: Theorist is non-procedural. The equals op (like all ops) states a relationship, and is not a command. You do not use the equals op to calculate the right-hand side of an equation and store the value in a variable on the left, as you would in many programming languages.

Relational

$x \neq 5$
$x < 5$
$x > 5$
$x \leq 5$
$x \geq 5$

Five Relational ops are supported in addition to the equal sign: $<$, $>$, $\leq$, $\geq$, and $\neq$. They evaluate to 1 or 0, depending on whether the proposition is true or false, respectively. Arguments can be integers or real or complex numbers.

Use Relational ops (except for $\neq$) to set limits on the value of real variables in working statements. For example, if x should never assume values outside the range -1 to $+1$, enter the inequalities "$x \geq -1$" and "$x \leq 1$", and make each of them working statements. These restrictions are used to create graphs and for other purposes.

Relational ops can also be used as constraints during automatic case generation. (See the Manipulations chapter.)

To enter the Relational ops, type "<" , and ">" from the keyboard. Use Option-=, Option->, and Option-< for ≠, ≥, and ≤ (as is the Macintosh convention). You can also enter these as "<>", ">=", and "<=" (as is the convention in Pascal).

Function
$\sin(x)$

The Function Call op is an invisible op that connects a function to its argument. You create a Function Call op by entering a function name and an opening parenthesis. A vector argument is treated as multiple arguments; type commas between arguments.

Do not create a function with the same name as a variable (e.g., $y = y(x)$). If a function always takes the same argument, state this as an equation between variables instead. For example, if you want to express the relationship $y(x) = \sin(x)$, and you never plan to use the function with any other argument (e.g., $y(z)$, or $y(t)$),write this as an equation in two variables: $y = \sin(x)$, rather than defining y as another name for the sine function.

If you raise a function to a power, enclose the function in parentheses. For example, write $[\sin(x)]^2$ rather than $\sin^2 x$. The function as a whole is raised to the power of the exponent. Similarly, use the function $\arcsin(x)$ rather than $\sin^{-1}(x)$. If you use a function of the form $\sin^2 x$ in an expression, the function converts to the more literal form $[\sin(x)]^2$ the next time you execute the Simplify manipulation. Expressions of the form $\sin^{-1}(x)$ simplify to $\arcsin(x)$.

Power
x^8

The Power op always has two operands, the base and the exponent. For integer exponents, the base is multiplied by itself a number of times as indicated by the exponent. The base can be any scalar or matrix (or any other operator, for symbolic operations), or any sum of such expressions.

If the exponent is an integer, the base can be of class Constant, Variable, or M- or D-Linear. If the exponent is not an integer, the base can only be of class Variable or Constant. If the exponent is a fraction or an irrational or complex number, the result can, in theory, have

several different values. However, Theorist returns only one, the principal value.

If the base is positive, and the exponent is real, the result is always positive. If the base is negative or complex, the result may also be negative or complex.

To enter x^8, type "`x^8`", or "`x**8`", or "`x` Command-H `8`".

Index or Subscript
x_8

The Index or Subscript op can be applied to a name, a matrix, or a vector. If applied to a matrix or a vector, it indicates an element or row. If applied to a name, it creates another name of the same class.

If you create the indexed name P_n, this name is independent of the names P and n. However, if P_n, has not been defined, Theorist uses any available information about P, possibly selecting the n-th element of P.

In most cases, the Index op is used for matrix and vector element selection. For example, if M is a matrix, then M_3 evaluates to the third row of M, a vector. The name $M_{(2,3)}$ refers to the element of M in the second row of the third column of M. If V is a row or column vector, then V_n selects the n-th element of V. If S is a scalar, then $S_{(m,n)}$ equals zero if $m \neq n$, or is the value S if $m = n$. (An indexed scalar is multiplied by an identity matrix.)

Index numbering starts at one. Indices that are non-positive, not real, or greater than the number of elements in the row or column of the matrix, yield either an error or the unknown value (?). Real-valued indices are rounded off to the nearest integer.

To enter M_3, type "`M_3`". Use Shift-minus or Command-L to enter the underbar.

Absolute Value
$|x|$

The Absolute Value op returns the absolute value of scalar arguments, the length of vector arguments, and the determinant of matrices.

To enter $|x|$, type "`|x`". Do not type the closing vertical bar. Alternatively, you can type "`abs(x)`".

Factorial
$x!$

The Factorial op can be applied to all scalars. For positive integers, it returns the product of all smaller integers down to 1. For other values it uses the Gamma function.

To enter $n!$, type "`n!`".

Square Root
$\sqrt{x}$

The Square Root op can be applied to all scalars. It is equivalent to the power op (see above) with an exponent of 1/2. If the argument is real and positive, it returns a positive result.

To enter $\sqrt{x}$, type "`\x `". You can use Command-R instead of the backslash key. As an alternative, type "`sqrt(x)`".

Adjoint
$x^\dagger$

The Adjoint op creates the transpose of a real matrix, the conjugate of a complex number, or the adjoint of a complex matrix (the transpose of the matrix and the conjugate of all elements). It has no effect on self-adjoint entities. However, in symbolic manipulations, it may not be apparent that a given entity is self-adjoint.

To declare that a given expression is real or self-adjoint, use a transformation rule such as:

upon [Simplify] transform $x^\dagger$ into x

To enter $x^\dagger$, type "`x` Option-3".

Matrix
$\begin{pmatrix} 1 & 2 \\ 3 & x \end{pmatrix}$

The Matrix op can hold one to 32,000 rows and one to 32,000 columns. A vector is a matrix with only one row or column, and can hold two to 32,000 elements. (In practice, the size is limited by available memory.) A one-by-one matrix cannot exist. Matrix and vector elements must be scalars for numeric manipulations. For symbolic manipulations, matrix and vector elements can be any object.

Matrix addition and multiplication work according to conventional rules. When a scalar is added to a matrix, the scalar is multiplied by an appropriate identity matrix. Scalars can only be added to square matrices.

Taking the reciprocal of a matrix with the division or power op yields the matrix inverse (only on square matrices). Taking the absolute value of a matrix yields the determinant of a square matrix, or the "length" of a vector (the square root of the sum of the squares of the components). Taking the adjoint of a matrix transposes it, and then takes the adjoint of all of its elements.

For symbolic manipulations, matrices can contain other matrices or other M- or D-Linear Operators as elements.

To enter:

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

type "`(1,2,3;4,5,6;7,8,9)`". Use commas to separate elements in a row, and semicolons to separate rows. To add rows to a matrix, insert the cursor after an element and type a semicolon.

Dot Product
x • *r*

The Dot Product op returns the pairwise product of the elements of two vectors. The two must have the same number of elements, but can be either row or column vectors. For example:

$$(a, b, c) \bullet (a, b, c) = a^2 + b^2 + c^2$$

The dot product of two vectors is independent of their orientation; either can be a row or column vector. The result is always a scalar.

In contrast, normal multiplication of two vectors is only allowed if they are of different types (one is a column vector, one is a row vector). If the first is a column vector, the result is the outer product of the two vectors (a matrix). If the first is a row vector, the result is the inner product (a scalar).

To enter *x* • *r*, type "x Option-8 r".

Cross Product
$x \times r$

The Cross Product op takes two 2-vectors or two 3-vectors (whether row or column) and returns the cross product. Vector elements can be of class Constant, Variable, or M- or D-Linear. The result is a scalar for 2-vectors or a 3-vector for 3-vectors. For example:

$$(a, b, c) \times (d, e, f) = (-ce + bf, cd - af, -bd + ae)$$

Choose **Expand** from the **Manipulate** menu to derive the symbolic expansion.

To enter $x \times r$, type "**x** Option-7 **r**".

Partial Derivative
$\frac{\partial}{\partial x}$

The Partial Derivative op is an abstract D-Linear Operator that takes a single variable and is multiplied onto the expression that it acts upon.

The partial derivative of a variable with respect to itself is one. If two variables are declared independent, the partial derivative of one with respect to the other is zero. (See the Independence Declaration in the Propositions chapter.)

To calculate an expression containing a partial derivative, select it and choose **Calculate** from the **Manipulate** menu. An increment value of 0.0001 is used unless you specify another. To specify an alternate increment value, create a working statement such as:

$$\boxed{\bullet}\ dx = 0.01$$

If you choose **Simplify** from the **Manipulate** menu, derivatives are processed symbolically.

To take the full derivative of an expression, multiply the predefined name *d* onto the expression. To do this, select an expression and type an opening parenthesis. This encloses the expression. Then click on the left most parenthesis and type "**d**".

To enter $\frac{\partial}{\partial x}$, type "Option-6 **x**".

Integral

$$\int_0^{-\pi} x^2\,dx$$

The Integral op takes one, two, or three arguments, depending on whether it has no bounds, or one or two bounds. Bounds can be infinite.

Most integral expressions contain some differential expression. However, if the integrand is zero, no differential is required.

To perform a symbolic integration, select the expression and choose either **Simplify** or **Expand** from the **Manipulate** menu.

To perform a numeric integration, select the expression and choose **Calculate** from the **Manipulate** menu. An integral must have upper and lower bounds to calculate its value.

To enter an integral with no limits, such as:

$$\int x dx$$

type "`$x*d*x`".

To enter an integral with limits, such as:

$$\int_1^2 x dx$$

type "`x*d*x` Option-4 `1` Tab `2`".

To enter an integral with only one limit, create an integral with two limits and delete one of the limits.

Summation

$$\sum_{k=1}^{100} a_k x^k$$

The Summation op takes as its three arguments a variable, and two integers for lower and upper bounds. Summations are D-Linear Operators. They are multiplied onto the expression that they act on.

The Summation op iteratively adds together instances of the expression it is multiplied onto as the variable increments from its lower to upper bound.

To numerically evaluate a summation, choose **Calculate** from the **Manipulate** menu. To symbolically expand a summation, choose **Expand** from the **Manipulate** menu.

To enter:

$$\sum_{k=1}^{100} a_k x^k$$

type "`k @ 1 Tab 100 Esc a_k Esc x^k`". It may be easier to enter summations from the palette.

Pi Product

$$\prod_{k=1}^{100} (x - a_k)$$

The Pi Product op takes four arguments: an iteration variable, two integers (for lower and upper bounds), and an expression as a main argument. The main argument should include at least one instance of the iteration variable. The Pi Product op iteratively multiplies together instances of the main argument, as the variable increments from its lower to upper bound.

The Pi Product op, unlike the Summation op, encloses its main argument rather than being multiplied onto it.

To evaluate a Pi Product, choose **Calculate** from the **Manipulate** menu. To symbolically expand a Pi Product, choose **Expand** from the **Manipulate** menu.

To enter:

$$\prod_{k=1}^{100} (x - a_k)$$

type "`(x-a_k)#k Tab 1 Tab 100`". It may be easier to enter Pi Products from the palette.

Conditional

$$\begin{cases} 0 & (x < 0) \\ x^2 & (x \geq 0) \end{cases}$$

The Conditional op takes an even number of arguments (four or more) arranged in a two-column matrix. It functions as a case statement in Pascal or C. The first column of the matrix contains possible return values and the second column contains conditional statements that determine which value is returned (if any). Use the Conditional op for constructing piece-wise continuous functions.

If the first conditional statement evaluates to True (nonzero), the operator returns the expression in the first column. If the first conditional statement evaluates to False (zero), the operator evaluates the second conditional statement, and so on, for as many rows as there are in the matrix.

If none of the conditional statements evaluate to True, the undefined value, a question mark, is returned.

To evaluate, choose **Calculate** or **Simplify** from the **Manipulate** menu.

To create a Conditional op press Option-Shift-[, and then insert a two-column matrix. It may be easier to enter a Conditional op from the palette. To add rows to the matrix, select the matrix and choose Add Row from the matrix pop-up menu on the palette. You can also insert a semicolon as described under the Matrix op.

Evaluate At

$\left[\frac{1}{3}x^3\right]_{x=1}^{x=-1}$

The Evaluate At op has two forms. The most common form contains two limits that evaluate an enclosed expression, using two substitutions, and subtracts the two results. It is used to evaluate definite integrals.

If you use only one limit, that value is substituted into the expression.

To evaluate, choose **Calculate** or **Simplify** from the **Manipulate** menu.

To enter:

$\left[\frac{1}{3}x^3\right]_{x=1}^{x=-1}$

type "Option-[`1/3` Esc `*x^3` Tab `x=1` Tab `x=-1`".

Range

$-1\ldots1$

The Range op represents a range of values, from minimum to maximum. Both limits must be integers or real numbers or names that evaluate to real numbers. The range op is only used in graph theories.

To enter $-1\ldots1$, type "`-1...1`". Enter at least two periods; extra periods are ignored.

Wildcard Variables 𝒶 to 𝓏

Wildcard variables are special symbols that can represent any expression. Each of the twenty-six letters of the alphabet can be used as a wildcard variable. A range of letters is available to match different types of expressions.

Wildcard Variables	Match
𝒶 - 𝒽	Any constant expression, except matrices
𝒾 - 𝓃	Any non-negative literal integer
𝑜 - 𝓆	Any expression, except functions
𝓇 - 𝓏	Any constant or variable expression

To enter 𝓍, type "?x" or use the variable palette's pop-up menu of wildcard variables.

There are three instances in which wildcard variables are used:

- The Substitute manipulation
- The Transform manipulation (and other transformation rules)
- Working statements of functions

With normal equations, you can Substitute expressions in one variable into different expressions with the same variable. For example, you can Substitute the equation $\sin(2x) = \cos(y) + \cos(z)$ into any equation that contains the target expression $\sin(2x)$. However, $\sin(2x)$ does not match the target expression $\sin(2\theta)$.

You can use wildcard variables to match target expressions of a given "pattern" or form regardless of the particular variables. If you create the trigonometry identity sin(2𝔭) = 2sin(𝔭)cos(𝔭), you can Substitute this relationship into any equation which contains a target expression of the form sin(2*anything*). For example, sin(2(*A* + *B*)) expands to 2sin(*A* + *B*)cos(*A* + *B*).

A pattern can contain more than one wildcard variable. If they are the same wildcard, they match only identical expressions. For example, the pattern 2sin(𝔵)cos(𝔵) does not match the expression 2sin(*θ*)cos(r), but does match 2sin(*θ*)cos(*θ*).

Each wildcard variable matches only a particular set of expressions. The expression sin(@𝔵) matches sin(5*x*) but not sin(*xy*). Wildcard variables @ through 𝔥 only match constants, not variables. Nor does sin(@𝔵) match sin(*x*) or sin(–*x*). Neither of these expressions contain a leading constant. In fact, sin(@𝔵) does not even match sin(–5*x*) because the negation op encloses the entire expression 5*x*.

Wildcard variables can be very particular or very indiscriminate. The expression sin(𝔭) matches the sine of any expression, positive or negative, simple or complex (except for function names not used as functions).

The wildcard variable 𝔵 has no relationship to the variable *x*. Do not use wildcard variables in name declarations.

For information on using wildcard variables with the Transform manipulation, see the Manipulations chapter.

For information on using wildcard variables with function statements, see the Propositions chapter.

Collapsars .....

Collapsars are abbreviations of long, unwieldy expressions. Symbolic algebra programs often produce huge, complicated equations that can stretch to several meters. In Theorist, you can compress any expression—large or small—to a "collapsar," which appears as an ellipsis (...).

To collapse an expression, select it, then choose **Collapse** from the **Notebook** menu (or press Command-\). To expand a collapsar, select it, then choose **Expose** from the **Notebook** menu (or press Command-'). Choose **Expose All** to expose all collapsars in the current theory.

Addition and multiplication expressions, when collapsed, behave a bit differently than other expressions. A collapsed string of terms (or factors) is usually displayed as two ellipses surrounding one term (or factor):

$$\cdots\cdots - 243y^5 + \cdots\cdots$$

If you select just an ellipsis, and choose **Expose** from the **Notebook** menu, you can "page" through the expression term by term. If the last term is displayed (on the left or right) only one ellipsis is shown:

$$32x^5 + \cdots\cdots$$

To expand the entire expression, select both ellipses and the displayed term. If you scroll to the first (or last) term, the preceding (or trailing) ellipsis is not shown.

If you choose **Auto Collapsar** from the **Prefs** menu, all new expressions wider than five and a half inches (14 cm) are displayed as collapsars.

To print out a large expression (without printing multiple pages sideways and without printing a collapsar that shows little or none of the underlying expression), Copy and Paste the expression into a comment to see its complete textual representation.

Parentheses ( )

Parentheses, brackets, and curly braces are used interchangeably in expressions to mark conceptual groups. For example:

$$(([(((\{x+7\}x+21]x+35)x+35\}x+21]x+7)x+1$$

Theorist does not distinguish between the three sets of grouping characters (( ), [ ], and { }). If you type "`[x+1)`", $(x + 1)$ is displayed. Unnecessary parentheses are usually deleted from expressions. If you want a specific set of parentheses for final output, export the equation into Expressionist and edit it there. See Appendix C, Using Expressionist with Theorist.

To group an expression in parentheses, select it and type an opening parenthesis.

Question Mark (?)

The question mark is used to signify a "blank space" when you are entering an equation. If left in an expression during a manipulation, it evaluates to an undefined value.

Each undefined value is independently undefined. For example, ? – ? does not evaluate to zero, but to ?. Zero times an undefined value is also undefined, rather than zero. Any arithmetic involving an undefined value results in the undefined value.

If a calculation or manipulation returns a question mark, some item in the manipulation is undefined or is dependent on an undefined value. For example, the Taylor series of $\sin(x)/x$ returns a question mark because the function has a removable singularity at $x = 0$. To get the Taylor series of $\sin(x)/x$, select $\sin(x)$, take the Taylor series of that, and then divide out the x by choosing **Expand** from the **Manipulate** menu.

Manipulations

When you work with Theorist, you enter equations, manipulate equations, and graph equations. This chapter discusses the various ways that you can manipulate your equations.

Some manipulations are very simple and are usually performed by hand (with the mouse). To perform more complicated manipulations, such as factoring a polynomial, you select an expression and then choose an operation from the **Manipulate** menu.

This chapter presents:

- An overview of manipulation options
  - Auto Simplify
  - Auto Casing
  - Arbitrary Constants
  - Manipulating in place
- Hand manipulations
  - Isolate
  - Move Over
  - Commute
  - Substitute
- Command manipulations
  - Calculate
  - UnCalculate
  - Simplify
  - Expand (and MiniExpand)
  - Collect
  - Factor
  - Apply
- Other manipulations
  - Transform
  - Taylor Series
  - Integrate by Parts

Manipulation Options

When you manipulate an expression, you set off a chain reaction that results in one or more new expressions logically derived from the original. Depending on different preferences that you set, new expressions can appear in different forms. In particular, you should be aware of the following ways you can affect the creation of new expressions:

- Auto Simplify
- Auto Casing
- Arbitrary Constants
- Manipulating in place

Auto Simplify

Simplify is a powerful, multifaceted manipulation that reduces the complexity of an expression. If Auto Simplify is on, all other manipulations (except Commute and Apply) simplify an expression just before displaying it. Auto Simplify executes the Simplify manipulation repeatedly (up to ten times) until the expression stops changing.

If you want to follow every step of a manipulation, turn Auto Simplify off. There are a few other instances when you will want Auto Simplify off. For example, the Collect manipulation "spreads out" the factors of a product, but Simplify groups them back together again. To rearrange the factors, turn Auto Simplify off and use Collect:

$$\frac{a^2b^3}{x^4y^5} = a^2b^3x^{-4}y^{-5}$$

Then use the Commute and Simplify manipulations to organize the factors differently:

$$\frac{a^2b^3}{x^4y^5} = a^2\frac{b^3}{y^5}\frac{1}{x^4}$$

Collect also factors out negative one (–1) from expressions, whereas Auto Simplify multiplies it back in.

All examples in this chapter assume that Auto Simplify is on, unless stated otherwise.

To turn Auto Simplify on or off, choose **Auto Simplify** from the **Prefs** menu (or press Command-2).

Auto Casing

Two manipulations, Isolate and Move Over, can produce more than one possible result. For example, there are two different solutions to $x^2 = 4$: $x = 2$ and $x = -2$. If Auto Casing is on, both cases are displayed in separate case theories. If not, only the primary case is displayed. If Auto Casing is on, up to sixteen separate case theories can be generated by a single manipulation. All examples in this chapter assume that Auto Casing is on, unless stated otherwise. (See Case Theories in the Propositions chapter.)

Also, if you integrate an expression (with Simplify, Expand, or Integrate by Parts), and Auto Casing is on, the manipulation can generate one or more arbitrary integer constants.

To turn Auto Casing on or off, choose **Auto Casing** from the **Prefs** menu (or press Command-3).

Arbitrary Constants

Some manipulations can generate one or more arbitrary constants. Real numbers are represented by the letter *c* and a numeric subscript. Integers are represented by the letter *n* and a numeric subscript. These subscripts start at 100 and are incremented by one for each new arbitrary constant. To change these defaults, choose **Arbitrary Constants** from the **Prefs** menu.

If Auto Casing is on, Isolate, Move Over, Simplify, and Expand can all produce arbitrary constants in the following situations.

Symbolically integrating an expression with Simplify or Expand can produce a real arbitrary constant. Using the Integrate by Parts manipulation can also produce a real arbitrary constant.

Isolating the argument of transcendental (e.g., trigonometric, exponential, and hyperbolic) functions produces arbitrary constants if Auto Casing is on. For example, if you use Isolate to find a solution in x for the equation $y = \tan(x)$, you create an arbitrary constant:

$$\square\ y = \tan(x)$$
$$\triangle\ x = \arctan(y) + n_{100}\,\pi$$

The constant n_{100} indicates that any *integer* creates a valid statement. That is, the following are possible solutions for x:

$x = \arctan(y) + 5\pi$

$x = \arctan(y) + -3\pi$

But not:

$x = \arctan(y) + \frac{3}{2}\pi$

Manipulating in Place

If you hold down the Command and Option keys when you execute a manipulation, the new expression (the conclusion) replaces the original. Otherwise, the conclusion appears on a new line below the original expression.

Hand (Mouse) Manipulations

The following manipulations are most easily (and most often) performed by hand:

- Isolate
- Move Over
- Commute
- Substitute

Simple manipulations, such as rearranging the terms of an expression, or moving a term of an equation from one side of the equals sign to the other, are easily performed by hand.

Three manipulations, Isolate, Move Over, and Commute, are very similar. In fact, the difference between executing one or another of these manipulations often depends on the exact spot where you release the mouse; a small distance to the left or right is all that distinguishes an Isolate manipulation from a Move Over manipulation.

To invoke any of these manipulations, select an expression (double-click on the expression or click-and-drag over the expression), then hold down the Command key. The mouse pointer turns into a pointing hand. Using this hand, click on the selected expression and drag it to a new location. Theorist highlights the different places where you can release the selection to perform a manipulation. Releasing the mouse performs the manipulation; which manipulation, if any, is determined by *where* you release the mouse.

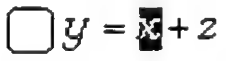
The Commute manipulation highlights possible insertion points with a small rectangle. The Move Over manipulation highlights one whole side of an equation and displays a thin line between the selection rectangle and the other side of the equation. The Isolate manipulation displays a large, highlighted rectangle beyond and outside the equation with a thin line connected to the selection rectangle.

Isolate

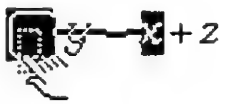
Use Isolate as a simple "solve for" manipulation. Select an expression inside an equation, and drag it with the pointing hand.

To Isolate an expression to the left, drag it over the proposition icon. To Isolate an expression to the right, drag it past the right-hand side of the equation. If you don't drag it far enough, you might execute a Commute or Move Over manipulation rather than an Isolate manipulation.

When you are in a position to make an Isolate manipulation, a thin bar appears connecting the selected expression and the release point.



Selected *x*



Isolate *x*

Once you've selected an expression, you can also execute the Isolate manipulation by choosing **Isolate** from the **Manipulate** menu. The menu manipulation always isolates an expression to the left.

Selected Expression	After Isolate Manipulation	Comment
$y = 5x + 7$	$x = \frac{1}{5}(y - 7)$	
$y = 5x + 7$	$5x = y - 7$	
$y = 5x + 7$	$5x + 7 = y$	
$y = 5x + 7$	$5 = \frac{y - 7}{x}$	Auto Casing off

Isolate does not always "solve" an equation if there is more than one occurrence of the given expression in your equation.

The Isolate manipulation can solve quadratic polynomials if you select *both* occurrences of the variable or expression to Isolate, and drag them left or right.

Isolate generates multiple case theories (with additional equations, if necessary) and equations with arbitrary constants, if necessary.

Selected Expression	After Isolate Manipulation	Comment
$\square\ x^2 - 3x = -2$	$\triangle\ x = \frac{1}{3}(x^2+2)$	Only one x isolated
$\square\ x^2 - 3x = -2$	$\circ\ \triangle\ x = 2$ $\circ\ \triangle\ x = 1$	Two case theories generated
$\square\ ab = c$	$\circ\ \square\ b \neq 0$ $\triangle\ a = \frac{c}{b}$ $\circ\ \square\ b = 0$ $\triangle\ 0 = c$	Two case theories generated
$\square\ \tan(x) = y$	$\triangle\ x = \arctan(y) + n_{100}\,\pi$	Generates arbitrary constant
$\square \begin{pmatrix} a & b \\ c & d \\ e & f \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix}$	$\triangle\ d = 4$	Isolating matrix element
$\square\ y = x^4$	$\circ\ \triangle\ x = y^{\frac{1}{4}}$ $\circ\ \triangle\ x = -y^{\frac{1}{4}}$ $\circ\ \triangle\ x = -iy^{\frac{1}{4}}$ $\circ\ \triangle\ x = iy^{\frac{1}{4}}$	
$\square\ y = x^5$	$\triangle\ x = e^{\frac{2}{5}n_{102}\pi i}\ y^{\frac{1}{5}}$	Exponents greater than four produce arbitrary constants

If Auto Casing is off in these situations, you only get the first answer, with no case theories and no arbitrary constants.

The Isolate manipulation does not work with multiple selections, except for solving quadratic equations as shown.

Move Over

Use the Move Over manipulation to relocate an expression from one side of an equation to the other. As with Isolate, select an expression, then press and hold the Command key. With the pointing hand, drag the expression to the other side of the equation. Do not drag as far as you would to execute an Isolate manipulation. Isolate and Move Over highlight in the same way: a bar is displayed between the selection and possible release points.

Once you have selected an expression, you can also execute a Move Over manipulation by choosing **Move Over** from the **Manipulate** menu.

Selected Expression	After Move Over Manipulation	Comment
$5x+3=y$	$5x=y-3$	
$5x=y-3$	$x=\frac{1}{5}(y-3)$	
$5x=y-3$	$0=-5x+y-3$	
$5x+3=y$	$5=\frac{y-3}{x}$	Auto Casing off
$4x-2=3x-(2-x)$	$0=0$	

The Move Over manipulation does not work with multiple selections.

Commute

Use the Commute manipulation to rearrange terms in a sum or factors in a product. Select one or more expressions, press and hold the Command key, and drag the expressions to the left or the right. If you drag an expression across the equals sign in an equation (or outside an equation), the manipulation is no longer a Commute, but a Move Over or Isolate manipulation.

When you click and drag terms of an expression with the pointing hand, all possible release points highlight. This is particularly useful when you are working with algebra and non-commutative operators (e.g., derivatives, summations). Only valid Commute operations highlight a release point. For example, if you try to move the x^2 in the following expression, only one other position is valid (to the left of π), whereas the 4 can move to three valid locations (each of which is highlighted in turn):

$$\square\ 4\frac{\partial}{\partial x}\pi x^2$$

Isolate and Commute highlight differently. Isolate displays a bar between the two highlighted rectangles, and Isolate is farther away from the equation:.

$$\square\ (x+y)^2 = x^2+y^2+2xy$$

Commute

$$\square\ (x+y)^2 = x^2+y^2+2xy$$

Isolate

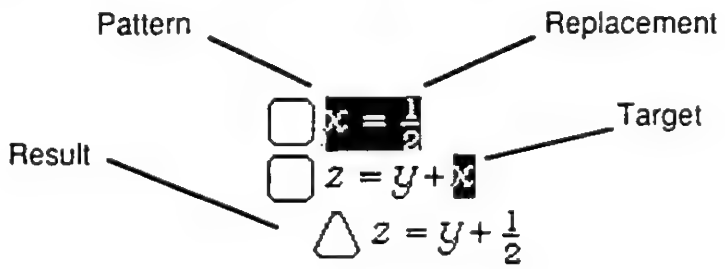
To reverse the order of a series of terms or factors, select the series, and choose **Commute** from the **Manipulate** menu. If you select only a single term (e.g., 5, x, $\sqrt{x+y}$), the menu manipulation has no effect. Commute as a hand manipulation allows you to rearrange individual terms and factors. When you Commute an expression using the menu option, the order of the terms is reversed.

Selected Expression	After Commute Manipulation	Comment
$3x^2$	$x^2 \cdot 3$	By hand
$3x^2+x^2-2x+5$	$5-2x+x^2+3x^2$	By menu choice
$3x^2-x^2-2x-5$	$3x^2+(-2x+x^2)+5$	By menu choice
$5x\frac{\partial}{\partial x}\sin(x)$	$5x\frac{\partial}{\partial x}\sin(x)$	Partial derivatives do not commute with their variables
$5x\frac{\partial}{\partial x}\sin(x)$	$x\frac{\partial}{\partial x}\sin(x)5$	Constants do commute in differential expressions

The Commute manipulation works with multiple selections, but only if you use the menu command.

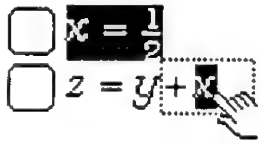
Substitute

The Substitute manipulation is a powerful technique for replacing expressions of a particular form with an equivalent expression of a different form. This manipulation utilizes a substitution equation and a target expression. The substitution equation consists of a pattern expression and a replacement expression.



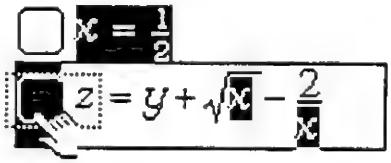
You can invoke the Substitute manipulation in three ways:

1. Select a substitution equation and drag it over a target expression to replace a single expression.

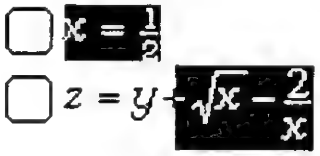


If you drag a substitution equation over an equation with an expression that matches the replacement expression (rather than the pattern expression), Substitute works in reverse. That is, the replacement expression is used as the pattern expression and vice versa. This reversal only occurs with this form of the Substitute manipulation.

2. Select a substitution equation and drag it over the icon of another equation to replace all occurrences of the target expression in that equation.



3. Select one or more substitution equations and one or more (non-equation) target expressions. Then choose **Substitute** from the **Manipulate** menu to replace all occurrences of each target expression.



Non-equations are always interpreted as targets, while equations are always interpreted as substitutions. To make an equation a target, select both sides of the equation individually, or select the proposition icon.

Note: You do not need to select target expressions with total precision. Selecting the square root of *x* will work as well as selecting just *x*.

To select a substitution equation, click on its equals sign. Use the pointing hand (press and hold the Command key) to drag the substitution equation into place. Target expressions highlight when they will accept a substitution. Release the mouse to execute the substitution.

If you use wildcard variables (e.g., x, y) in a substitution equation, they match any expression of the type associated with that wildcard variable. Substitution expressions with regular variables only match those explicit variables. To enter wildcard variables, type a question mark, then the variable letter (e.g., "?x"). For more information, see Wildcard Variables in the Expressions chapter.

The targets and substitution equations must be in the same theory, or in nested theories. The result is displayed in the most deeply nested theory.

If you find you are performing certain substitution manipulations frequently, you may want to create Transform manipulations to handle these replacements. The Transform manipulation is described later in this chapter.

Command Manipulations

Command manipulations are invoked by choosing an option from the **Manipulate** menu. Like hand manipulations, the result is displayed below the selected expressions, unless the manipulation is executed in place. To replace the selected expression (with the new expression), hold down the Command and Option keys while executing the manipulation.

This section discusses the following manipulations:

- Calculate
- UnCalculate
- Simplify
- Expand (and MiniExpand)
- Collect
- Factor
- Apply

Calculate generates a numeric evaluation of an expression. Simplify and Expand reduce expressions to simpler, more fundamental forms, whereas Collect and Factor tend to create more compact expressions. With Apply you can change an expression in many ways while maintaining its logical integrity.

All command manipulations, with the exception of Apply, can be used on multiple selections.

All examples in this section assume that Auto Simplify is on, unless stated otherwise.

Calculate

Calculate tries to evaluate an expression to a number like a pocket calculator. It may also produce a matrix or vector of numbers. Calculate uses working definitions of names (if necessary). If an expression cannot be fully calculated, because of undefined names or for other reasons, Calculate evaluates all possible subexpressions.

The results of a Calculate manipulation are usually accurate to the full nineteen digits of precision, if possible. You can control the number of digits displayed by choosing the desired accuracy from the **Prefs** menu's **Display Precision** sub-menu. The full nineteen digits are maintained internally.

Selected Expression	After Calculate Manipulation	Comment
$3-\frac{9}{2}$	−1.5	
4π	12.566	Reverse with UnCalculate
sin(1)	0.84147	
$\frac{1}{8}x+7$	$0.125x+7$	If x is undefined
$\frac{1}{8}x+7$	7.375	If $x=3$
y	7.375	If $x=3$ and $y=\frac{1}{8}x+7$

To calculate the partial derivative of an expression with respect to a particular variable, that variable must have an assigned value. The increment value used for the independent variable is determined by a working statement of the form $dx = 0.001$. In the absence of such a definition, Theorist uses $1/10{,}000^{th}$ of the difference between a defined minimum and maximum (e.g., $x > 0$ and $x \leq 100$). If neither is available, an increment value of 0.0001 is used, regardless of the value of the variable.

Definite integrals (i.e., integrals with both upper and lower limits) are calculated numerically. The limits can be any real values including infinity and negative infinity. An integrand must contain a differential to indicate the variable of integration. Integral calculation uses an internally defined increment value. The variable of integration must be a real number.

Selected Expression	After Calculate Manipulation	Comment
$\frac{\partial}{\partial x}y$	$\frac{\partial}{\partial x}y$	If x and y are undefined
$\frac{\partial}{\partial x}y$	0.125	If $x=3$ and $y=\frac{x}{8}+7$
$\frac{\partial}{\partial x}\sin(1)$	0	
$\int_0^1 x^3 dx$	0.25	
$\int_{-\infty}^{\infty} \frac{dx}{1+x^2}$	3.1416	

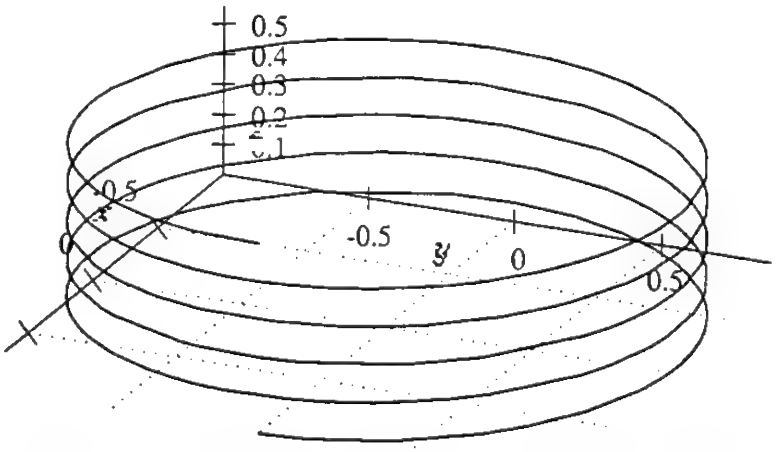
If you want to integrate over a path in space, or over the complex plane, parametrize the path over some real variable and calculate the integral over that variable. For example, consider the spiral curve described by the following working statements:

$x = \cos(5t)$

$y = \sin(5t)$

$z = \frac{t}{10}$

Using these statements you can create a Line plot in a three dimensional graph theory:



Using the Pythagorean theorem, you can calculate the length, *s*, of the curve:

$$\square\ (ds)^2 = (dx)^2 + (dy)^2 + (dz)^2$$

With the help of some of the rules in "Standard Rules and Declarations," (on your distribution disks), you can calculate the integral for this path in space:

$$\triangle \int_0^{2\pi} ds \quad = 31.422$$

For more information about creating graphs of this type, see the Graph Programming chapter.

Before graphing a polynomial with algebraic coefficients (e.g., $x^3 + 3\sqrt{2}\,x^2 + 6x + 2\sqrt{2}$), use Calculate to evaluate the terms. This improves graphing speed.

To calculate an expression, select it, and choose **Calculate** from the **Manipulate** menu (or press Command-5).

UnCalculate

The UnCalculate manipulation attempts to undo the effects of the Calculate manipulation. If Calculate produces a number such as 6 or 42, UnCalculate has no effect. However, UnCalculate can recreate simple fractions and constants from decimals. UnCalculate reduces to zero "very small" factors and terms (10^{14} times smaller than other values in an expression).

UnCalculate also returns square and cube roots of rational numbers, products of π or π^2 and rational numbers, and e raised to rational powers.

Selected Expression	After UnCalculate Manipulation
0.3333333333333333	$\frac{1}{3}$
12.56637061435917295	4π
20.08553692318766773	e^3

Note: For UnCalculate to regenerate an original value, selected values must be exact to approximately ten digits of precision for the Macintosh II version and to approximately 13 digits of precision for the regular version of the program (regardless of the selected display precision).

Simplify

Simplify executes a wide range of operations designed to reduce the complexity of an expression. Simplify combines constants, terms, and factors using rational arithmetic, cancels where possible, and arranges expressions into canonical form.

Products are sorted in class order where possible (Constants, Variables, M-Linear Operators, D-Linear Operators), with constants to the left. Sums are sorted in class order, with constants to the right. Imaginary components are sorted so they appear after real components. Powers and subscripts are sorted in numerical order, to arrange polynomials by descending powers. In the absence of other criteria, Simplify sorts in alphabetical order.

Constants are reduced to simple forms, if possible. Powers of powers are distributed out. Derivatives are evaluated if possible, as are simple integrals of linear functions, and linear functions of simple integrals.

Selected Expression	After Simplify Manipulation	Comment
$\sqrt{x}\sqrt{x}-\frac{x^2}{x}+\frac{6}{2\cdot 3+0}-1$	0	
$\frac{1}{\sqrt{2}}-(x-1)$	$-x+\frac{1}{2}\sqrt{2}+1$	
$5+x^2+x^3+x\cdot 2+\frac{x^{10}}{2}$	$\frac{1}{2}x^{10}+x^3+x^2+2x+5$	
$x^3y+2x^2y^2+xy^3$	$xy^3+2x^2y^2+x^3y$	
$e^{2\pi i}+\log(1000)$	4	
$(x^2)^n$	x^{2n}	Reverse with Collect (if Auto Simplify is off)
$\int (x^4+\sin[x])\,dx$	$-\cos(x)+\frac{1}{5}x^5+c_{100}$	Generates an arbitrary constant (c_{100})
$\begin{pmatrix} A & B \\ C & D \end{pmatrix}_{[1,2]}$	B	
$\frac{(-x-1)(x-2)}{x+1}$	$-x+2$	

The Simplify manipulation also:

- Propagates negations through powers

$$(-x)^5 \rightarrow -x^5$$

$$\sqrt{-x} \rightarrow i\sqrt{x}$$

- Removes negations within absolute values

$$|-x| \rightarrow |x|$$

- Adjusts function notation to canonical form

$$\sin^2 x \rightarrow (\sin(x))^2$$

$$\sin^{-1} x \rightarrow \arcsin(x)$$

- Invokes working order statements

$$x^5 \rightarrow 0$$ (if $x^5 = 0$ is a working statement)

- Compresses functions of anti-functions

$$\sin(\arcsin(x)) \rightarrow x$$

- Interprets element of identity matrix

$$5_{(3,3)} \rightarrow 5$$

$$5_{(3,2)} \rightarrow 0$$

- Distributes the Adjoint op

$$(x \sin(y))\dagger \rightarrow x\dagger \sin(y\dagger)$$

- Moves fractional powers of integers out of denominators

$$\frac{1}{\sqrt{3}} \rightarrow \frac{1}{3}\sqrt{3}$$

- Evaluates the Evaluate At op
- Executes conditional ops if the conditional expression can be calculated

If the Auto Simplify option is on (choose **Auto Simplify** from the **Prefs** menu), a Simplify manipulation is executed after every manipulation (except Commute and Apply). This is the default. Turn off Auto Simplify if you want to proceed through a manipulation step-by-step, or if you want to rearrange the parts of an un-simplified expression.

Auto Simplify may cause problems when working with integrals. If Auto Simplify is on, the integrand is simplified before figuring the integration. In most cases, this works fine. Occasionally, however, the simplified expression is more difficult to integrate. For example, consider the following integral:

$$\int d\sin(\sin(x))$$

If Auto Simplify is on, Simplify produces the following solution:

$$\int \cos(x)\cos(\sin(x))\,dx$$

If Auto Simplify is off, you get:

$$\sin(\sin(x))$$

Simplify also applies all transformation rules listed in the current theory as "Upon Simplify." (See the Transform manipulation later in this chapter.)

To simplify an expression, select it and choose **Simplify** from the **Manipulate** menu (or press Command-1). You can also select an expression, hold down the Command key, and double-click on the expression. To simplify an expression in place, select it, hold down the Command key and Option key, and double-click on the expression.

Expand and MiniExpand

Expand executes a range of operations designed to increase the complexity of an expression. In many cases it reverses the Collect and Factor manipulations. During its operation, Expand may execute one or more Simplify manipulations.

MiniExpand only works on the outside layer of an expression. Expand is recursive in that it applies a MiniExpand manipulation to all expressions nested within the selected expression. However, Expand is not repetitive. Repeated Expand manipulations may further expand an expression.

Expand: $([x+1]^2+1)^2 \rightarrow x^4+4x^3+8x^2+8x+4$

MiniExpand: $([x+1]^2+1)^2 \rightarrow (x+1)^4+2(x+1)^2+1$

Expand symbolically distributes products, and multiplies out powers of sums, matrices, and derivative and summation operators. It expands matrix sums, inverses and determinants, dot products, and powers of products.

If applied to a rational function (one polynomial divided by another), Expand divides polynomials or distributes a common divisor over a sum. If applied to a fraction with an expanded polynomial in the numerator and a product of polynomials in the denominator, Expand performs a partial-fraction decomposition.

Selected Expression	After Expand Manipulation	Comment
$2x(3y+7)$	$14x+6xy$	Reverse with Collect
$x(x+1)^2(x-1)^2$	x^5-2x^3+x	Reverse with Factor
$\begin{pmatrix}3&4\\5&6\end{pmatrix}^2\begin{pmatrix}x\\y\end{pmatrix}$	$\begin{pmatrix}4[5x+6y]+3[3x+4y]\\6[5x+6y]+5[3x+4y]\end{pmatrix}$	Needs second Expand (see next)
$\begin{pmatrix}4[5x+6y]+3[3x+4y]\\6[5x+6y]+5[3x+4y]\end{pmatrix}$	$\begin{pmatrix}29x+36y\\45x+56y\end{pmatrix}$	

(Continued)

Selected Expression	After Expand Manipulation	Comment
$\begin{pmatrix} 3 & 4 \\ 5 & 6 \end{pmatrix}+\begin{pmatrix} x & y \\ z & 0 \end{pmatrix}+\frac{a}{2}$	$\begin{pmatrix} \frac{1}{2}a+x+3 & y+4 \\ z+5 & \frac{1}{2}a+6 \end{pmatrix}$	
$\frac{a}{2}\begin{pmatrix} x & y \\ z & 0 \end{pmatrix}$	$\begin{pmatrix} \frac{1}{2}ax & \frac{1}{2}ay \\ \frac{1}{2}az & 0 \end{pmatrix}$	Reverse with Collect
$\left(\frac{\partial}{\partial x}\right)^3 x^{100}$	$970200\,x^{97}$	
$\left(\frac{\partial}{\partial x}+x\right)^2 \sin[x]$	$2x\cos[x]+x^2\sin[x]$	
$\sum_{k=6}^{10} x^k$	$x^{10}+x^9+x^8+x^7+x^6$	
$\prod_{k=1}^{4} x^k$	x^{10}	
$\begin{pmatrix} 3 & 0 \\ y & 7 \end{pmatrix}^{-5}$	$\frac{1}{4084101}\begin{pmatrix} 16807 & 0 \\ -4141y & 243 \end{pmatrix}$	
$\left\|\begin{pmatrix} x & y \\ z & a \end{pmatrix}\right\|$	$ax-yz$	Determinant
$\frac{1}{4}\left(\frac{\partial}{\partial x},\frac{\partial}{\partial y},\frac{\partial}{\partial z}\right)\bullet\left(x^4,y^4,z^4\right)$	$x^3+y^3+z^3$	

(Continued)

Selected Expression	After Expand Manipulation	Comment
$(x^2y^3z^8)^5$	$x^{10}y^{15}z^{40}$	Reverse with Factor
$\frac{x^4+2x^3+5x^2-x+1}{x^2+1}$	$\frac{-3x-3}{x^2+1}+x^2+2x+4$	Reverse with Collect
$\frac{x+7}{x^2-1}$	$\frac{7}{x^2-1}+\frac{x}{x^2-1}$	Reverse with Collect
$\frac{x+7}{[x+1][x-1]}$	$-3\frac{1}{x+1}+4\frac{1}{x-1}$	Reverse with Collect

The Expand manipulation also performs the following actions:

- Distributes powers over products and quotients

$$\left(\frac{ab}{c}\right)^7 \rightarrow \frac{a^7b^7}{c^7}$$

- Distributes the absolute value op

$$\left|\frac{ab}{c}\right| \rightarrow \frac{|a||b|}{|c|}$$

- Distributes logs of products

$$\log(abc) \rightarrow \log(a)+\log(b)+\log(c)$$

- Distributes an integral over matrix elements

$$\int\begin{pmatrix} x & y \\ a & b \end{pmatrix} \rightarrow \begin{pmatrix} [\int x] & [\int y] \\ [\int a] & [\int b] \end{pmatrix}$$

Expand also applies all transformation rules listed in the current theory as "Upon Expand." (See the Transform manipulation later in this chapter.)

To Expand an expression, select it and choose **Expand** from the **Manipulate** menu (or press Command-6).

Collect

Collect separates out common terms of a sum and orders the remaining terms as a polynomial in a given variable, grouping coefficients by powers in the polynomial variable. Applied to a sum of fractions, Collect adds the fractions together and collects all terms over a common denominator. Collect also separates out common factors of a matrix.

Selected Expression	After Collect Manipulation	Comment
x^5	$xxxxx$	Auto Simplify off; Reverse with Simplify
$7x^4+21x^3+21x^2+7x$	$7(x^3+3x^2+3x+1)x$	Reverse with Expand
$xy^3z+x^3y^2z^2+x^2y^7z^3$	$(y+xy^5z^2+x^2z)xy^2z$	Reverse with Expand
$xy^3z+x^3y^2z^2+x^2y^7z^2$	$(y+[x^2+xy^5]z)xy^2z$	Reverse with Expand
$\frac{1}{x}+\frac{1}{y}+\frac{1}{z}$	$\frac{xy+xz+yz}{xyz}$	Reverse with Expand
$xy^3z+\frac{x^3y^2}{z^2}+x^2y^7z^3$	$\frac{(x^2+xy^5z^5+yz^3)xy^2}{z^2}$	Reverse with Expand
$\begin{pmatrix}12x^2 & 0\\ 9x^6 & 6x\end{pmatrix}$	$3x\begin{pmatrix}4x & 0\\ 3x^5 & 2\end{pmatrix}$	Reverse with Expand
$\frac{\partial}{\partial x}y+\frac{\partial}{\partial x}z$	$\frac{\partial}{\partial x}(y+z)$	Reverse with Simplify

Collect "spreads out" factors in a product so you can rearrange and regroup them in a new order. (Turn Auto Simplify off, to use Collect in this way.) For example, you can use the following manipulations to rearrange a fractional product:

$$\square\ \frac{3}{4}\frac{x^2y}{az^3}$$

Collect Expression

$$\triangle\ \frac{3}{4}\frac{x^2y}{az^3} = 3\cdot 2^{-2}\boxed{x^2}ya^{-1}z^{-3}$$

Commute x^2 by hand

$$\triangle\ \frac{3}{4}\frac{x^2y}{az^3} = 3\cdot 2^{-2}ya^{-1}\boxed{x^2z^{-3}}$$

Simplify part of expression

$$\triangle\ \frac{3}{4}\frac{x^2y}{az^3} = 3\cdot 2^{-2}ya^{-1}\boxed{\left(\frac{1}{z^3}x^2\right)}$$

Simplify again

$$\triangle\ \frac{3}{4}\frac{x^2y}{az^3} = \boxed{3\cdot 2^{-2}ya^{-1}}\frac{x^2}{z^3}$$

Simplify other part

$$\triangle\ \frac{3}{4}\frac{x^2y}{az^3} = \boxed{\left(3\,\frac{1}{4}\,\frac{1}{a^1}y\right)}\frac{x^2}{z^3}$$

Simplify again

$$\triangle\ \frac{3}{4}\frac{x^2y}{az^3} = \left(\frac{3}{4}\frac{y}{a}\right)\frac{x^2}{z^3}$$

Auto Simplify must also be off in order to use Collect to distribute power sums ($x^{a+b} \rightarrow x^a x^b$) and integer powers ($x^3 \rightarrow xxx$).

The polynomial variable determines the grouping of terms when you invoke the Commute manipulation. You can specify the polynomial variable of an expression with more than one variable by arranging the terms of the polynomial. The right-most variable of the first term is used as the polynomial variable. The following two equations are equivalent; in the first, x is the polynomial variable, in the second, y is the polynomial variable:

$$\square\, x^2+x+x^2y+xy+y+1$$
$$\triangle\, x^2+x+x^2y+xy+y+1 = (y+1)x^2+(y+1)x+y+1$$

$$\square\, x^2y+x^2+x+xy+y+1$$
$$\triangle\, x^2y+x^2+x+xy+y+1 = x^2+x+(x^2+x+1)y+1$$

Collect also applies the inverse of all transformation rules listed in the current theory as "Upon Expand." (See the Transform manipulation, later in this chapter.)

To Collect an expression, select it and choose **Collect** from the **Manipulate** menu (or press Command-7).

Factor

Factor executes several operations designed to dismantle an expression into its most fundamental terms. Integers are converted into their prime factors, and sums are factored as polynomials, if possible. Products with common factors are joined together. In general, Factor breaks down an expression more completely than Collect. For example, Collect separates out the common terms of a polynomial whereas Factor determines roots of a polynomial. In most cases, you can reverse the effects of Factor with Expand.

Factor can manipulate all polynomials with numeric coefficients (one variable), and some polynomials with symbolic coefficients (or polynomials with multiple variables). Quadratic polynomials are factored symbolically using the quadratic equation.

Selected Expression	After Factor Manipulation	Comment
-720	$[-1]2^4 3^2 5$	Reverse with Expand or Simplify
$-x^6$	$(-1)x^6$	
x^2-1	$[x+1][x-1]$	Reverse with Expand
x^2y^2	$[xy]^2$	Reverse with Expand
$x^2-11x-26$	$(x+2)(x-13)$	Reverse with Expand

If you select a cubic or quartic polynomial for factoring, a dialog box asks whether you want to proceed numerically or symbolically. Theorist uses the Cubic formula and the Quartic formula to symbolically factor third-order and fourth-order polynomials. These manipulations can be extremely time consuming, especially if the intermediate expressions do not cancel. On the other hand, numerically factoring such polynomials is usually very fast.

Fifth order and higher polynomials are always factored numerically. The roots are often real or complex numbers. These numbers are usually accurate to 17 digits.

You can specify the polynomial variable of an expression with more than one variable by arranging the terms of the polynomial. The right-most variable of the first term is used as the polynomial variable. For example, $x^2+2xy+y^2$ is factored in terms of x, whereas $y^2+2xy+x^2$ (or $2xy+y^2+x^2$) is factored in terms of y.

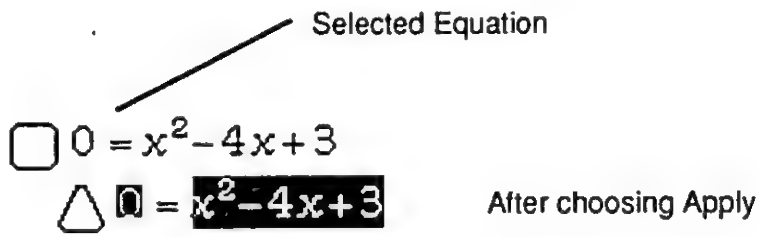
To Factor an expression, select it and choose **Factor** from the **Manipulate** menu (or press Command-8).

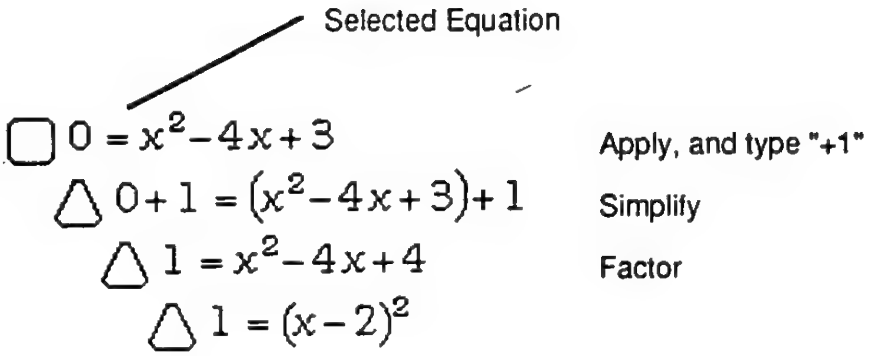
Apply

Apply is a generic manipulation that provides a technique for doing almost anything to both sides of an equation, or both the numerator and denominator of a fraction. Apply can also be used to multiply an expression by a term and its inverse (n and 1/n), or add and subtract the same value from an expression (+n and –n). Apply never changes the validity of an expression or equation, just the structure.

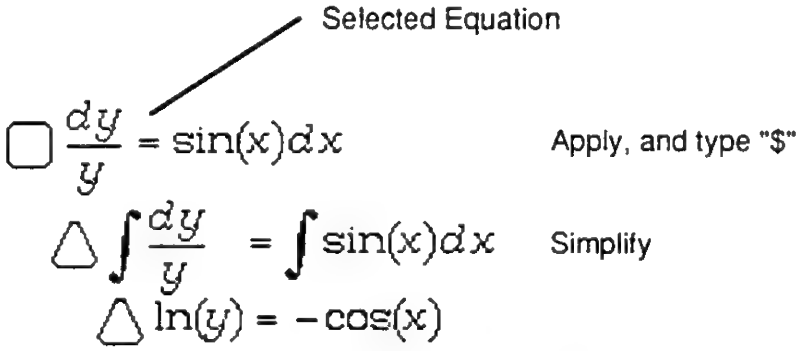
To use Apply, select an expression or equation and choose **Apply** from the **Manipulate** menu (or press Command-zero). There are four modes for using Apply, but in each instance, the manipulation creates a conclusion with two selected empty expressions (indicated by question marks).

Mode 1. Using Apply with an equation. In this mode, you can add or subtract the same expression from both sides of an equation. For example, to complete the square of a polynomial, select the equation (click on the equals sign), invoke the Apply manipulation, type "+1", then use Simplify and Factor:

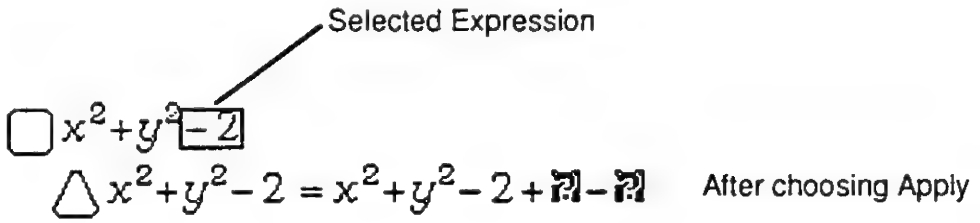


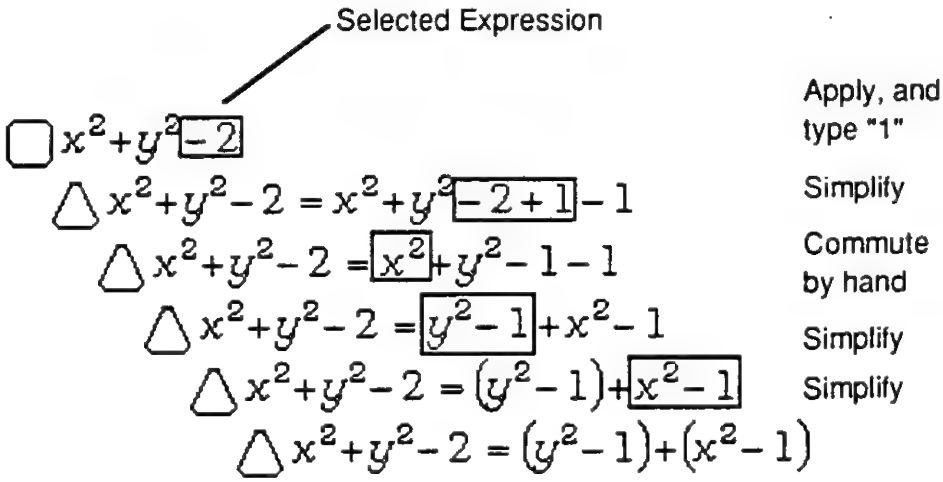


As another example of using Apply with an equation, consider calculating the integral of the following expressions. Select the equation, invoke Apply, type "$" (to enter an integral), then use the Simplify manipulation:

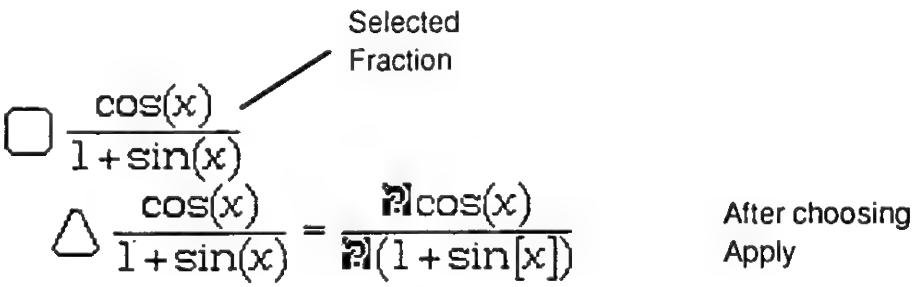


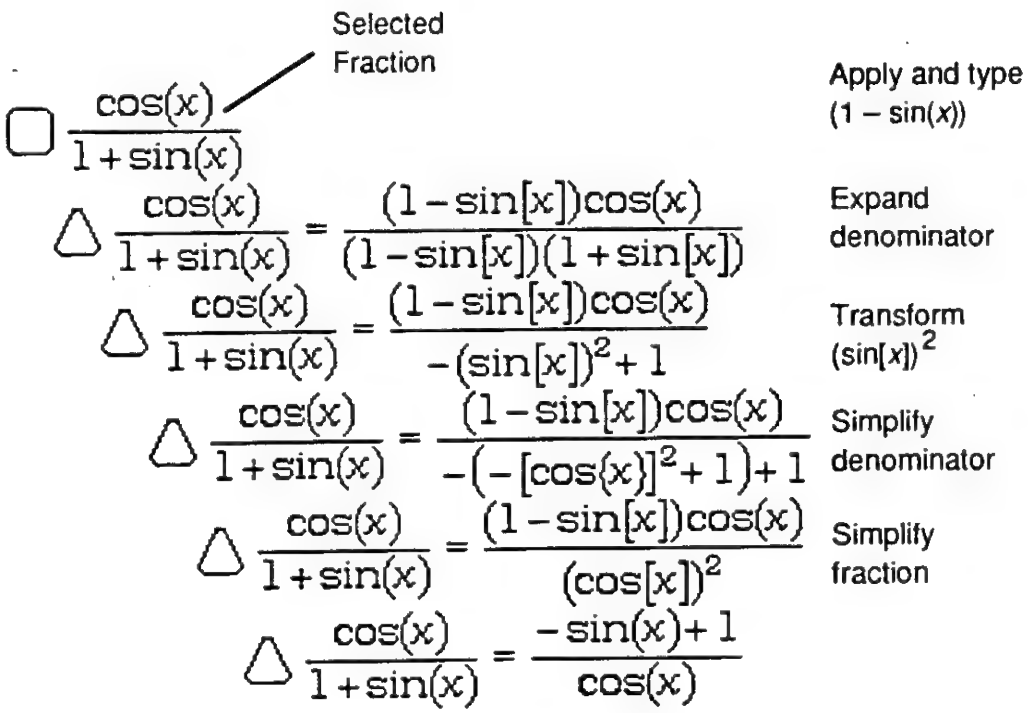
Mode 2. Using Apply with a single term. In this mode, you can add *and* subtract the same expression from a selected term. For example, you can turn –2 into two –1s:





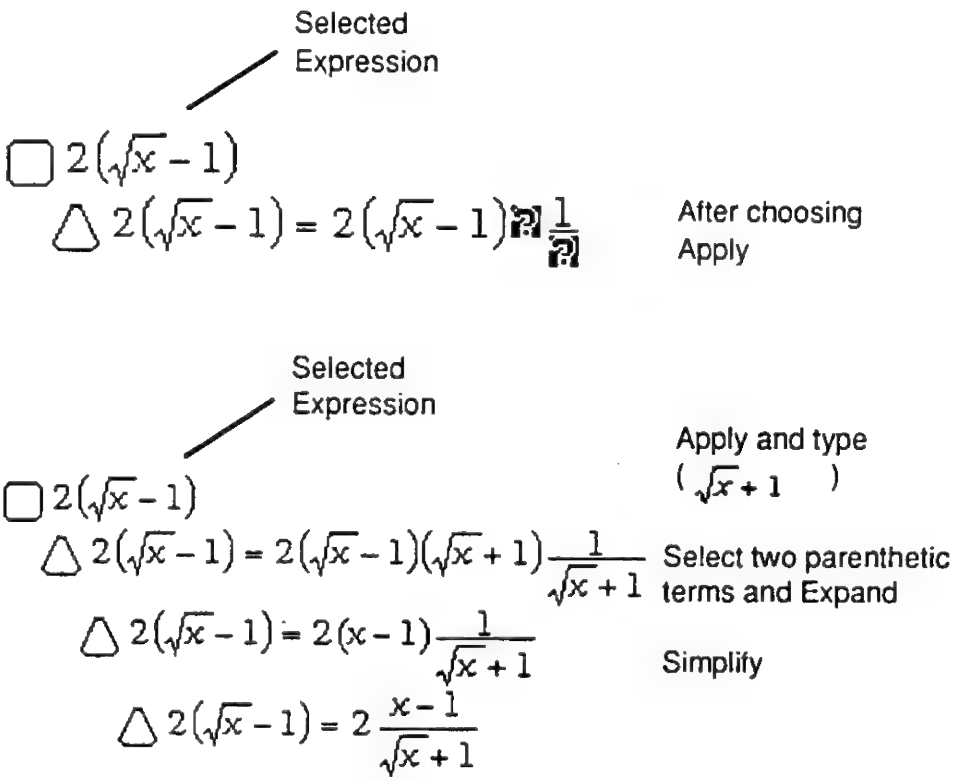
Apply mode 3. Using Apply with a fraction. In this mode you can multiply both the numerator and denominator of a fraction by the same expression. In the following example, you need to have copied in the bundle named "Standard Rules and Declarations," in order for the Transform manipulation to work as shown. Standard Rules and Declarations is one of the notebooks included on your distribution disks.





Note: When you use the Transform manipulation, a dialog box offers two possible transformations, accept the simpler form.

Apply mode 4. Using Apply with a product. In this mode you can multiply a term by an expression and the inverse of that expression.



Apply does not work with multiple selections.

Other Manipulations

Three other manipulations are also available on the Manipulate menu:

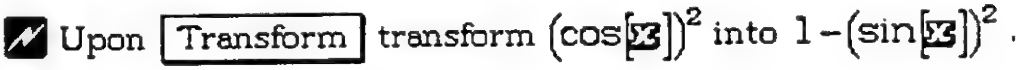
- Transform
- Taylor Series
- Integrate by Parts

The Transform manipulation allows you to create your own manipulation rules and to extend the functionality of existing manipulations (Simplify, Expand, and Collect). The Taylor Series manipulation creates a polynomial approximation of a function, and Integrate by Parts is useful for solving difficult integrations.

Transform

The Transform manipulation is designed as a means for you to extend Theorist's capabilities. Transform does not have any effect until you explicitly create a transformation rule that executes "Upon Transform."

Transform seeks to identify expressions of particular forms and turn them into different forms. For example, a transformation rule such as:



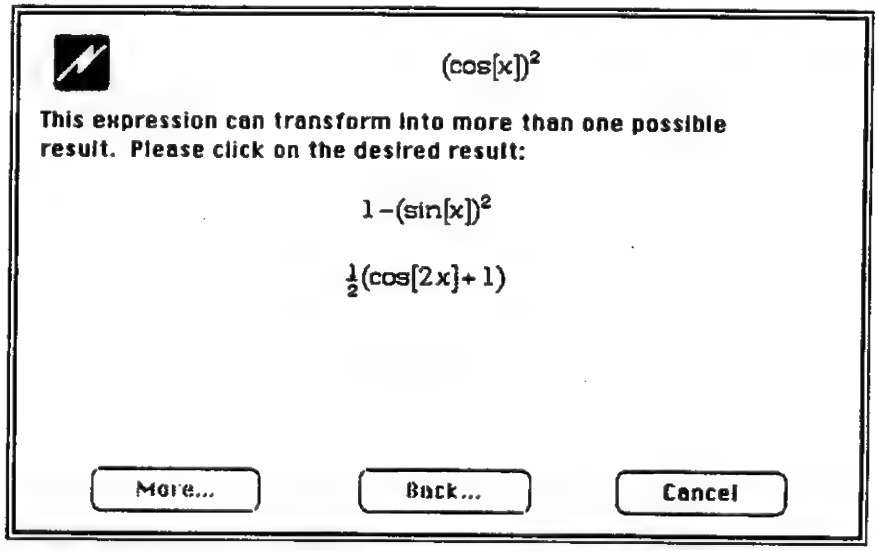
turns any selected expression of the form $(\cos [x])^2$ into an expression of the form $1 - (\sin [x])^2$, where x can represent almost any expression.

Use wildcard variables (e.g., x, y) to represent expressions in transformation rules. Wildcard variables match any variable of a given type. The type of variable is determined by the letter used. (See the Wildcard Variables section in the Expressions chapter.)

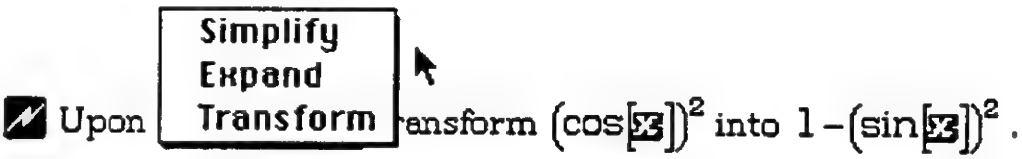
A Transform manipulation works very much like a Substitute manipulation except that the substitution equation is not an equation. The substitution equation is contained in the transformation rule. You select only the target expression, rather than the target *and* the substitution equation.

When invoked, the Transform manipulation consults all transformation rules in the current theory, as well as all parent theories, for any matching expressions. If a match is found, the transformation is carried out.

If more than one rule matches a target expression, a dialog box displays your choices:



You can also make transformation rules execute during Simplify, Expand, and Collect manipulations. Choose the desired manipulation from the rule's pop-up menu:



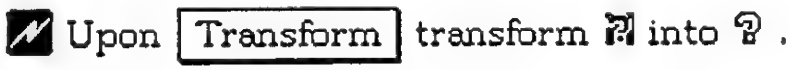
Collect reverses the action of transformation rules that are set to execute "Upon Expand."

In some situations, more than one transformation rule that executes with the Simplify or Expand manipulation will match a selected expression. In these cases, rather than displaying a dialog box, the manipulation executes the first matching rule. The manipulation searches the current theory from top to bottom, then the parent theory from top to bottom, and so on.

Transformation rules that execute "Upon Simplify" are carried out *after* executing the Simplify manipulation, whereas transformation rules that execute "Upon Expand" are carried out *before* executing the Expand manipulation.

The Transform manipulation is not recursive. It executes only once on the outermost layer of an expression. If nested expressions can still be transformed, repeat the manipulation. To make a Transform manipulation recursive, change the pop-up menu to Simplify or Expand. Both of these manipulations execute recursively.

To create a transformation rule, select an equation and choose **Transform Rule** from the **Input** menu (or press Command-Y). You can edit a transformation rule as you would any other proposition. If no equation is selected, a transformation rule is created with two question marks. Select the question marks and enter the appropriate expressions:



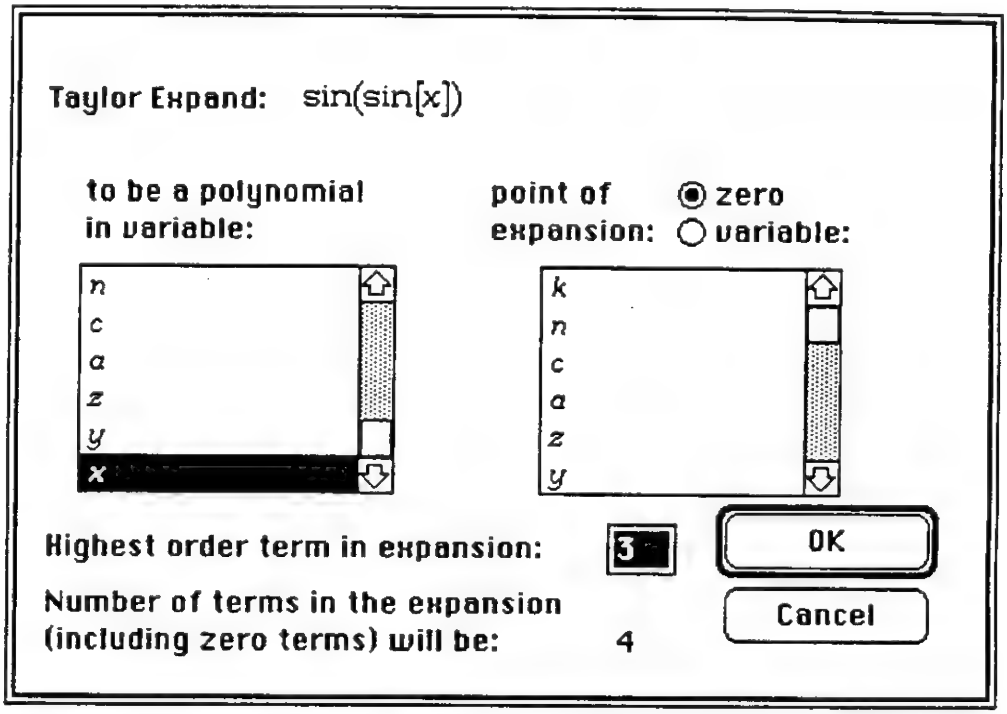
Taylor Series

The Taylor Series manipulation is specifically designed to generate the Taylor series of a continuous function. A Taylor series is a polynomial that approximates such a function. A single point of the function, usually zero, is designated as the expansion point. At that point, the Taylor series approximation is exact. The approximation deteriorates as you move away from that point. The higher the order of the polynomial, the better the approximation.

A Taylor series can have terms up to the 99th order (e.g., x^{99}). However, computing a Taylor series of this complexity may be inordinately time-consuming.

The expansion point for a Taylor series can be zero, another constant, or a variable. If you define it as a variable, you can generate distinct Taylor expansions in alternate case theories.

To create a Taylor series, select a function and choose **Taylor Series** from the **Manipulate** menu. A dialog box asks you to verify the variable to use, the highest order term in the expansion, and the point of expansion. You can create several Taylor expansions at the same time by selecting multiple functions and then choosing **Taylor Series**. The Taylor Series manipulation presents the following dialog box:



A Taylor series is generated by taking repeated derivatives of an expression, once for each order. Therefore, it is reasonable to generate an arbitrarily large Taylor series for polynomials, certain trigonometric functions (sin and cos), certain hyperbolic functions (sinh and cosh), and some exponential expressions (e^x). But most functions, including some trigonometric functions (tan, sec, tanh, sech), rational functions (a fraction with polynomials in the numerator and denominator) algebraically explode.

To create a multidimensional Taylor series, declare the variables independent, select an expression, and create a Taylor expansion once for each variable.

For example, to create a Taylor series in two dimensions, use the following steps. Create an equation where one variable is a function of two independent variables:

$$\boxed{\bullet}\ z = 1.5^{x}\sin(y)$$

Declare the variables x and y independent of each other:

The variables (x,y) are independent of [each other] .

Taylor expand the right-hand side of the equation twice. For the first one, select x as the polynomial variable (in the left-hand scroll box of the dialog box). For the second, designate y as the polynomial variable. If you choose third-order expansions (the default), the Taylor series approximation of the expression is:

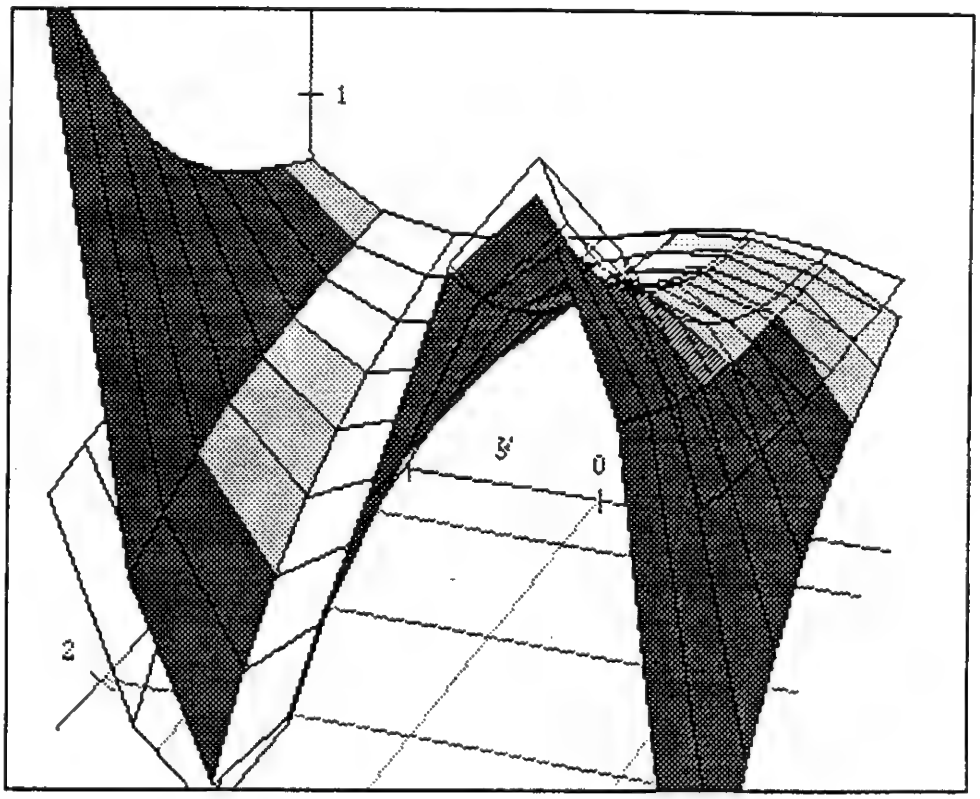
$$z = \left(-\tfrac{1}{6}y^3 + y\right) + \tfrac{1}{6}(\ln[1.5])^3 x^3\left(-\tfrac{1}{6}y^3 + y\right) + \tfrac{1}{2}(\ln[1.5])^2 x^2\left(-\tfrac{1}{6}y^3 + y\right) + \ln(1.5)\, x\left(-\tfrac{1}{6}y^3 + y\right)$$

To graphically compare the original equation to the Taylor expansion, set the approximation equal to z', and create a graph with a Surface plot for each equation. To create a graph as shown in the following figure, create an Illuminated 3D graph from the original equation. Then select the expansion equation and choose **Add Surface Plot** from the **Graph** menu. Finally, adjust the first plot proposition to display a transparent surface, so that the propositions appear as:

Surface at (x,y,z) where x = west ... east and y = south ... north ;
[Transparent] surface has [mesh] and is shaded using [Solid] coloring;
[White] is the solid color.

Surface at (x,y,z') where x = west ... east and y = south ... north ;
[Illuminated] surface has [mesh] and is shaded using [Solid] coloring;
[White] is the solid color.

The graph itself, with two Surface plots, should appear as follows. Note how the Taylor series approximation (shown in gray) is exact at (0, 0), but veers away as x and y increase.



Integrate by Parts

The Integrate by Parts manipulation is a useful technique for integration. If an integrand is the product of two expressions, one of which is easy to integrate, and the other is easy to differentiate, integration by parts can lead to a solution. The rule it uses is:

$$\int u dv = uv - \left(\int v du \right)$$

When you integrate by parts, you always get another integral, but the new integral may be easier to solve or may provide some leverage for solving the original integral.

To use the Integrate by Parts manipulation, select the *dv* part of the expression, and choose **Integrate by Parts** from the **Manipulate** menu (or press Command-4).

Integrate by Parts is commonly used where you have a low power of x multiplied by a transcendental function that is easy to integrate.

For example, consider the following integral:

$$\int x\sin(x)\,dx$$

Select $\sin(x)dx$ and integrate by parts:

$$\int x\sin(x)\,dx = -x\cos(x) + \sin(x)$$

As another example of using Integrate by Parts, consider the following integration:

$$\int x\sin(x)\,dx = -x\cos(x) + \sin(x)$$

To find a solution for this integral, select $\sin(5x)\ dx$, and integrate by parts:

$$\square \int \sin(3x)\sin(5x)dx$$

$$\triangle \int \sin(3x)\sin(5x)dx = -\tfrac{1}{5}\cos(5x)\sin(3x) + \tfrac{3}{5}\left(\int \cos[5x]\cos[3x]dx\right)$$

Commute the two cosine terms of the integral on the right, so that you can select $\cos(5x)dx$, and integrate by parts again:

$$\triangle \int \sin(3x)\sin(5x)dx = -\tfrac{1}{5}\cos(5x)\sin(3x) + \tfrac{3}{5}\left(\int \cos[3x]\cos[5x]dx\right)$$

$$\triangle \int \sin(3x)\sin(5x)dx = -\tfrac{1}{5}\cos(5x)\sin(3x) + \tfrac{3}{5}\left(\tfrac{1}{5}\cos[3x]\sin[5x] + \tfrac{3}{5}\left[\int \sin(5x)\sin(3x)dx\right]\right)$$

Expand the entire right-hand side, and use the Move over manipulation to consolidate the integral on the left-hand side:

$$\triangle \int \sin(3x)\sin(5x)dx = \frac{3}{25}\cos(3x)\sin(5x) - \frac{1}{5}\cos(5x)\sin(3x) + \frac{9}{25}\left(\int \sin[5x]\sin[3x]dx\right)$$

$$\triangle \frac{16}{25}\left(\int \sin[5x]\sin[3x]dx\right) = \frac{3}{25}\cos(3x)\sin(5x) - \frac{1}{5}\cos(5x)\sin(3x)$$

Solve for the integral by selecting the entire integral expression (click on the integral sign) and dragging it over the proposition icon. Expand the right-hand side one more time to see the relationship more clearly:

$$\triangle \int \sin(5x)\sin(3x)dx = \frac{25}{16}\left(\frac{3}{25}\cos[3x]\sin[5x] - \frac{1}{5}\cos[5x]\sin[3x]\right)$$

$$\triangle \int \sin(5x)\sin(3x)dx = \frac{3}{16}\cos(3x)\sin(5x) - \frac{5}{16}\cos(5x)\sin(3x)$$

Graphs

Graphing mathematical relationships is a powerful technique for visualizing and understanding the interactions of functions and variables. In Theorist, a graph is a two-dimensional space (or a representation of a three-dimensional space) for drawing plots. You can also animate any graph and display a family of related plots in a short movie.

This chapter discusses the fundamentals of two- and three-dimensional graphs and their plots. The next chapter, Graph Programming, describes how to modify graph theories and plot propositions to create graphs of your own design.

This chapter presents:

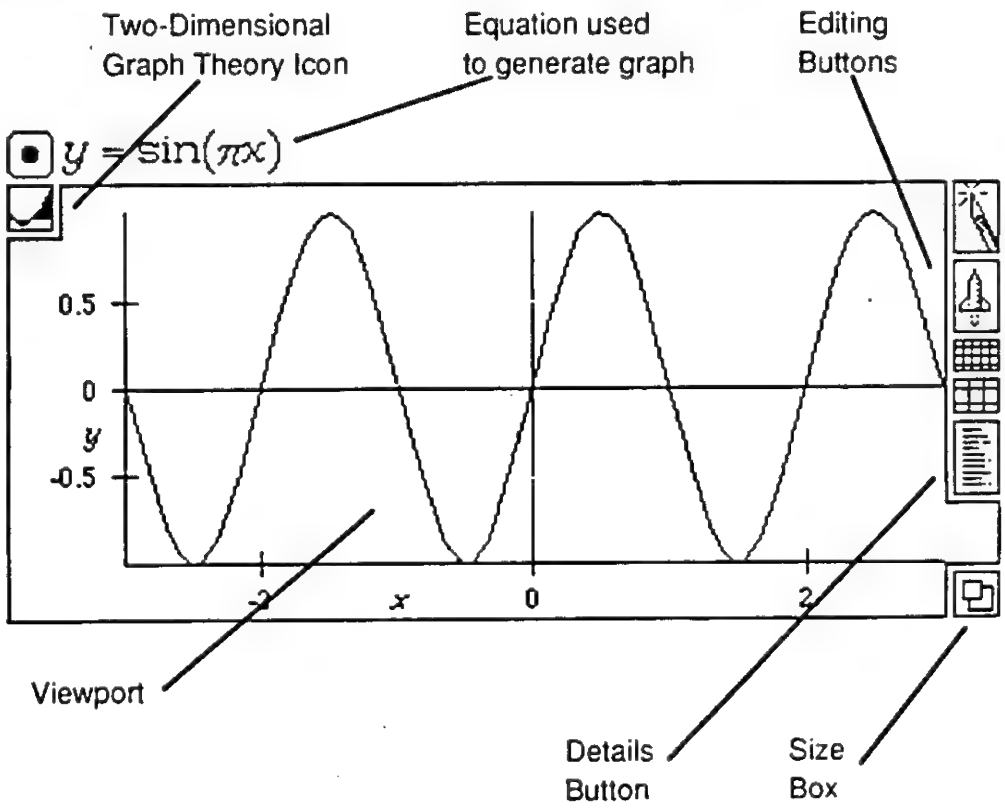
- Graph theories
- Two-Dimensional graph theories
- Three-Dimensional graph theories
- Animation
- Printing and Exporting

Graph Theories

A graph theory provides a framework, or shell, for displaying Plot propositions. When you create a graph, you actually create a graph theory and a set of associated plot propositions contained within the graph theory. All definitions and settings within a graph theory (e.g., boundary values, plot resolution) are true only for that graph. Graphs, like case theories, often look outside for information when necessary, but if a sufficient definition is available internally, external definitions have no effect.

This section introduces two- and three-dimensional graph theories and the different graphs that create them.

The following figure displays an equation and a two-dimensional graph created from the equation.

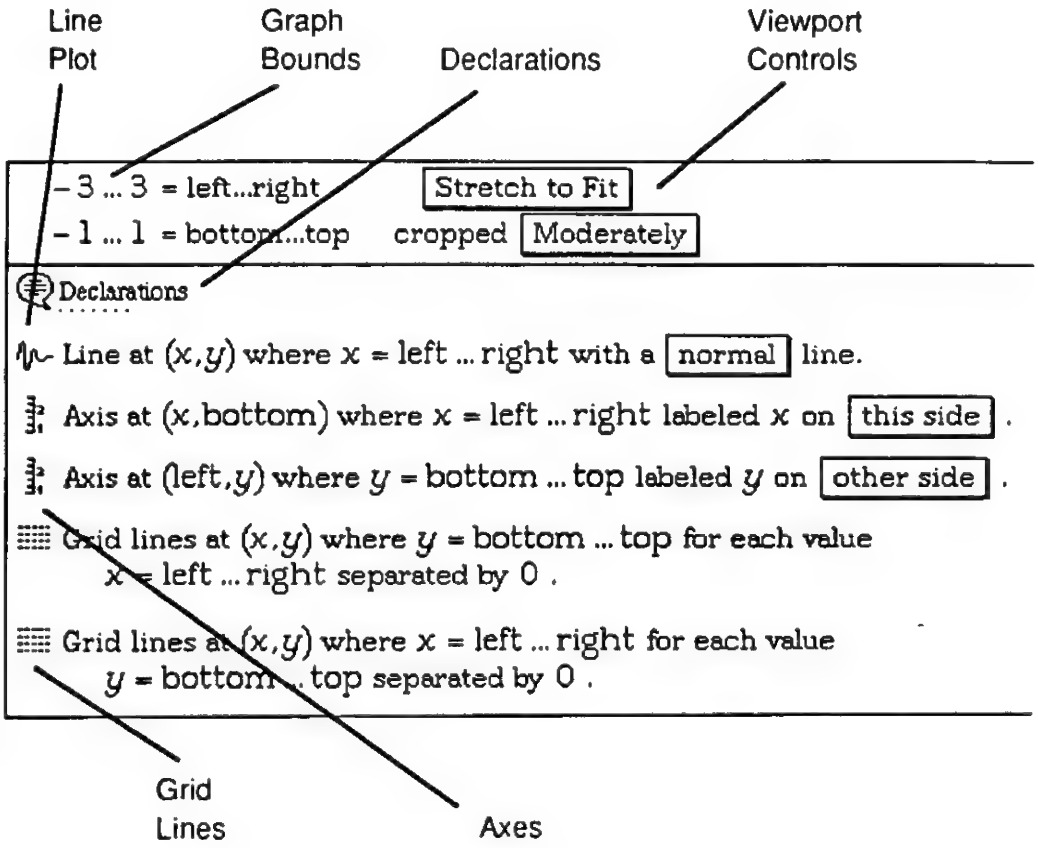


Each graph theory (displayed with an icon representing a two- or three-dimensional graph) consists of a viewport, a set of editing buttons, a details button, and a size box. If you click on the size box, the size of the viewport is displayed in points, inches, and millimeters. If you click and drag the size box, these values change dynamically. Click on the details button to open a box showing the graph details. Each statement in the details box is a proposition describing some element of the graph. You can alter the display of a graph with the editing buttons, or by entering values directly into the details box.

Graph details can contain any kind and any number of propositions. However, graph details usually contain information about the viewport, declared definitions, plots, axes, and grids. If you click on the details button, a box opens displaying the graph details.

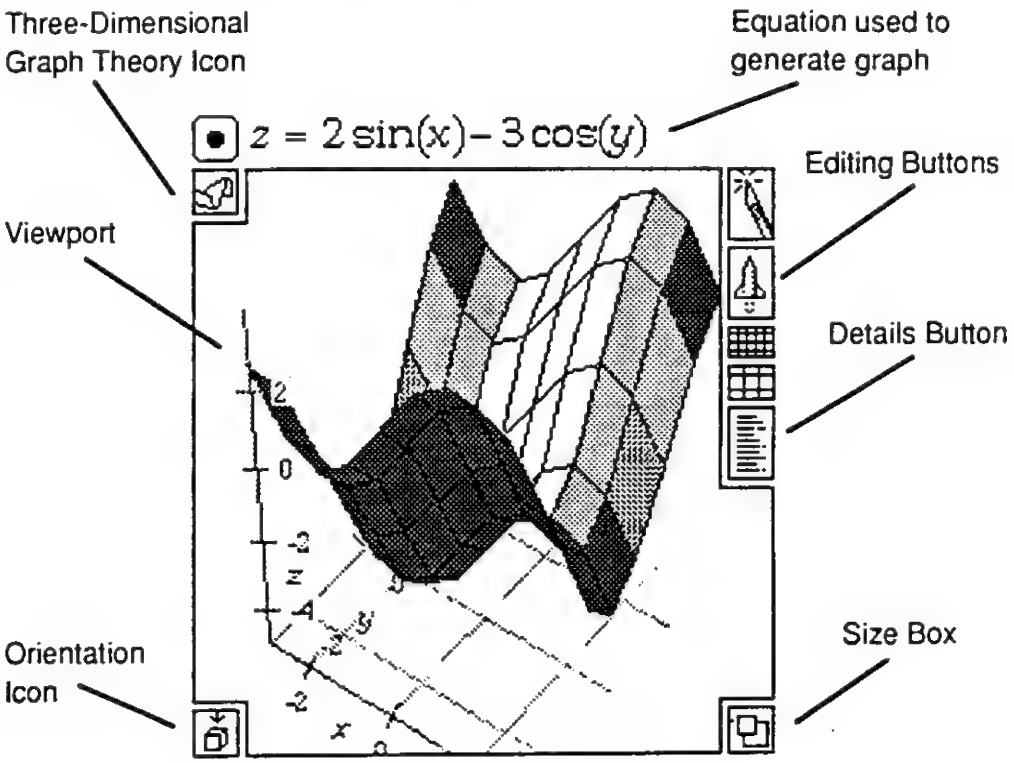
The details are like the fold-down control panel on a television. You can quickly create a graph and easily adjust your view with the editing buttons and the open-hand pointer, but if you want to fine tune a graph's appearance, adjust the details.

The graph details for a simple two-dimensional graph with a single Line plot:



Two-dimensional and three-dimensional graph theories are very similar. Three-dimensional graph theories include an *orientation* icon in the lower left-hand corner, which indicates the side of the graph shown in the viewport. Three-dimensional graph theories most often display three-dimensional plots (e.g., surfaces), although they can also display Line plots.

A three-dimensional graph theory:



Plots in a graph theory are propositions. They are displayed in a graph's viewport. Plot propositions include Line plots, Surface plots, and Contour plots, as well as the Axes and Grid lines.

When you point to the viewport with the mouse, the arrow becomes an open-hand. Use this pointer on two- and three-dimensional graphs to click-and-drag the displayed image. This is the most direct way to

change your view of a graph. The editing buttons at the right of the viewport also give you quick ways for changing your view. They perform the following functions:

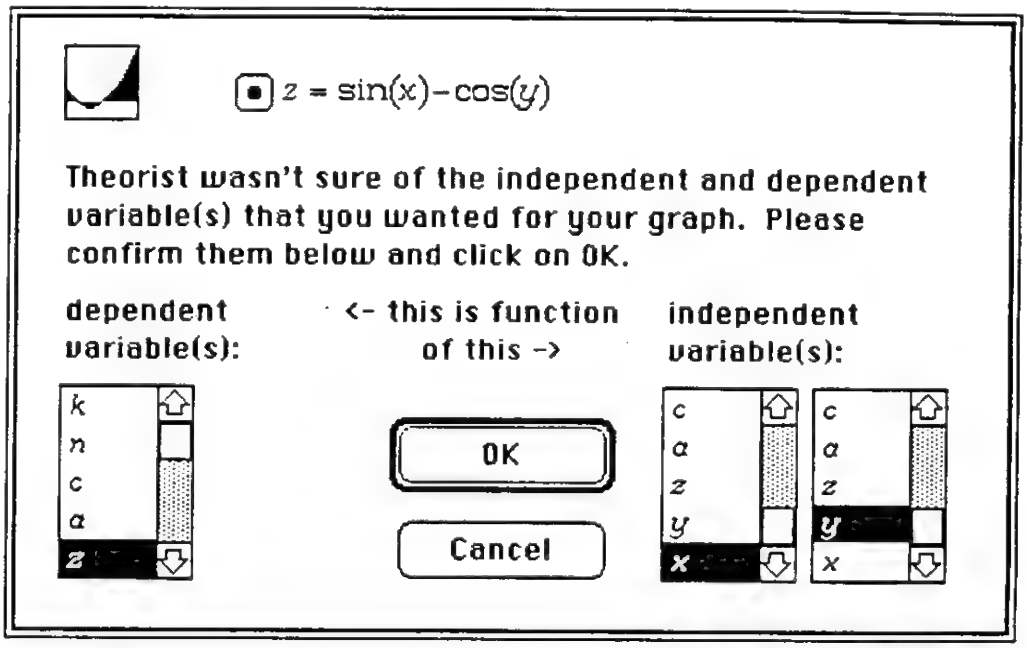
	Knife	Cut out a small portion of a plot to view it in greater detail.
	Rocket	Zoom out (or in, using the Option key) by a factor of two.
	Higher Resolution	Increase the number of data points (to create a smoother plot).
	Lower Resolution	Decrease the number of data points (to create a rougher but faster-drawn plot).
	Details	Show (or hide) the graph details.

Plots, Axes, and Grid lines are all graphic objects drawn into a graph's viewport. The details box contains the propositions for each of these objects as well as a set of definitions used by the graph. These propositions, like all Theorist propositions, are not programming commands; their order does not affect their function. Each is an independent description of a graphical object.

Creating Graph Theories

To create a graph theory, enter an equation (or equations, to create a parametric plot) that specifies a relationship between variables. Then choose the type of plot you want to use to display this relationship. The first three items on the Graphs menu provide options for creating twelve different types of plots. Seven of these are displayed in two-dimensional graph theories, and five in three-dimensional theories.

It's important to determine which variables should be *independent* (ranging freely between the boundary values) and which should be *dependent* (functions of the independent variables). A dialog box often appears when you create a graph asking you to designate the dependent and independent variables:



Select the independent variable(s) from the scroll box(es) on the right, and the dependent variable(s) from the scroll box(es) on the left. The first three options on the Graphs menu let you plot the relationship between:

- Two dependent variables and one independent variable
- One dependent variable and one independent variable
- One dependent variable and two independent variables

If any names in the current notebook have not been defined, you'll be asked to declare them.

x = f(t), y = g(t)
2 Dependent
1 Independent

Use the first item on the Graphs menu to construct a graph that displays the relationship between three variables where two are functions of a third variable (two dependent variables, and one independent variable). The result is a two-dimensional parametric graph.

y = f(x)
1 Dependent
1 Independent

Use one of the sub options of the second menu item to construct a graph that displays the relationship between two variables where one is a function of the other (one independent variable, and one dependent variable). There are four such graphs: Linear, SemiLog, Polar, and Complex 3D.

Graph	Graph Theory	Description
Linear	2 D	The independent variable is displayed on the horizontal (*x*) axis and the dependent variable is displayed on the vertical (*y*) axis. Both scales are linear.
SemiLog	2 D	The independent variable is displayed on the horizontal (*x*) axis (linear) and the dependent variable is displayed on the vertical (*y*) axis (logarithmic).
Polar	2 D	The independent variable is displayed along the circumference (θ) of a circle and the dependent variable is displayed along the radius (*r*) of the circle.
Complex 3D	3 D	The real component of the independent variable is displayed along the West-East axis and the imaginary component is displayed along the South-North axis. The absolute value of the dependent variable is displayed along the Bottom-Top axis and the complex phase is displayed as the surface color.

z = f(x, y)
1 Dependent
2 Independent

Use the third menu item to construct a graph that displays the relationship between three variables where one is a function of the other two (two independent variables, and one dependent variable). There are seven such graphs: Density 2D, Contour 2D, Zero Contour, Color 3D, Illuminated 3D, Spherical 3D, Cylindrical 3D.

Graph	Graph Theory	Description
Density 2D	2 D	The independent variables are displayed along the horizontal (*x*) and vertical (*y*) axes. The dependent variable is displayed as a white-to-black shading (white = 0, or less; black =1, or more)
Contour 2D	2 D	The independent variables are displayed along the horizontal (*x*) and vertical (*y*) axes. The dependent variable is displayed as a collection of labeled contour lines.
Zero Contour	2 D	The independent variables are displayed along the horizontal (*x*) and vertical (*y*) axes. The dependent variable is displayed as a single contour line where it equals zero.
Color 3D	3 D	The independent variables are displayed along the West-East and South-North axes. The dependent variable is displayed as a surface along the Bottom-Top axis using a specified range of colors depending on the elevation.
Illuminated 3D	3 D	The independent variables are displayed along the West-East and South-North axes. The dependent variable is displayed as a surface along the Bottom-Top axis. The surface is solid white illuminated with a light source over your left shoulder.

(Continued)

Graph	Graph Theory	Description
Spherical 3D	3D	The independent variables are displayed as the latitude, measured from the top rather than the equator (θ), and longitude (φ). The dependent variable is displayed as the distance from the center origin.
Cylindrical 3D	3D	The independent variables are displayed as the longitude (θ) and height. The dependent variable is displayed as the radius (r).

Because graphs theories *are* theories, they can contain other theories. See the file "Spherical Family" on the Theorist distribution disks for an example of using case theories within graph theories.

Two-Dimensional Graphs

Two-dimensional graphs consist of a two-dimensional graph theory (the shell) and one or more two- or three-dimensional plots (Line, Surface, or Contour) and associated Axes and Grid lines. This section presents:

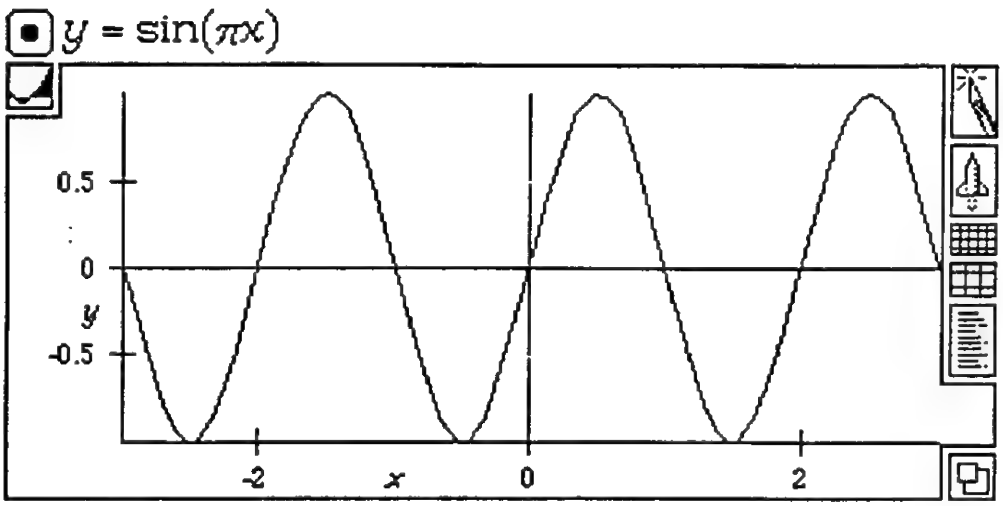
- Icons for editing two-dimensional graph theories
- Graph details of two-dimensional graph theories
- Descriptions of Line plots and Contour plots
- A list of graphs that create two-dimensional graph theories
- A technique for finding roots of a Line plot or two Contour plots

Once created, plot propositions can be adjusted, edited, and moved around like any other propositions. To move a plot, select it by its proposition icon and Cut (or Copy) and Paste it into another graph theory. Plot propositions are meaningless outside of a graph theory. Axes and Grid lines are also plots, and can be moved and edited in the same way.

You can move plots between two-dimensional graph theories without difficulty, using the Copy and Paste commands. However, it's often easier to select an equation and choose the appropriate menu option on the Graphs menu to add plots, axes or grid lines to an existing graph theory.

To add a plot (or axis or set of grid lines) to an existing graph theory, select an equation (and a graph theory if there is more than one), and choose **Add Line Plot**, **Add Surface Plot**, or **Add Contour Plot** (or **Add Axes** or **Add Grid Lines**) from the **Graphs** menu.

The most direct way to change your view of a two-dimensional graph theory is to use the mouse. When the pointer is over the viewport, it turns into an open-hand icon. Click, drag, and release the image as if it were a piece of paper; after a moment the image will redraw with new boundary values. You can also use the editing icons or change values in the graph details box to change the view on your graph.



Editing Icons

The Knife

Click on the Knife icon to *pick up* the knife (the cursor becomes a knife). Click and drag across a rectangular portion of the graph to look at just that part. The portion of the graph you select can be most of the viewport or just a tiny portion. Using the Knife changes the values of the graphs bounds.

The Rocket

Click on the Rocket icon to *zoom out* (from the center of the viewport) by a factor of two. Press Option and click on the Rocket to *zoom in* by a factor of two. After a moment, the graph redraws. To zoom in or out by more than a factor of two, click the icon repeatedly. To zoom out by a factor of 1000, click ten times. To view a small portion of the graph, it's faster to use the Knife. Zooming in and out is reflected in the values of the graph bounds.

The Resolution Icons

Click on the dense grid icon to increase the data points used to create the plots in the graph by (approximately) a factor of two. Click on the open grid icon to (approximately) halve the number of data points. The more data points used, the smoother the plot line(s), and the longer it

takes to generate the image. Plot resolution is not reflected in the graph details.

When you click on a resolution icon, the current value and the destination value appear briefly to the left of the icon (e.g., 8 -> 16). Resolution values are 1, 2, 4, 8, 16, 32, 64, and 128 (default: 8). For Line Plots, Axes, and Grid Lines, the value indicates the approximate number of line segments used to create a 90-degree curve. For Surface Plots and Contour Plots, the value indicates the number of panels on each side of the displayed plot.

The Details Box

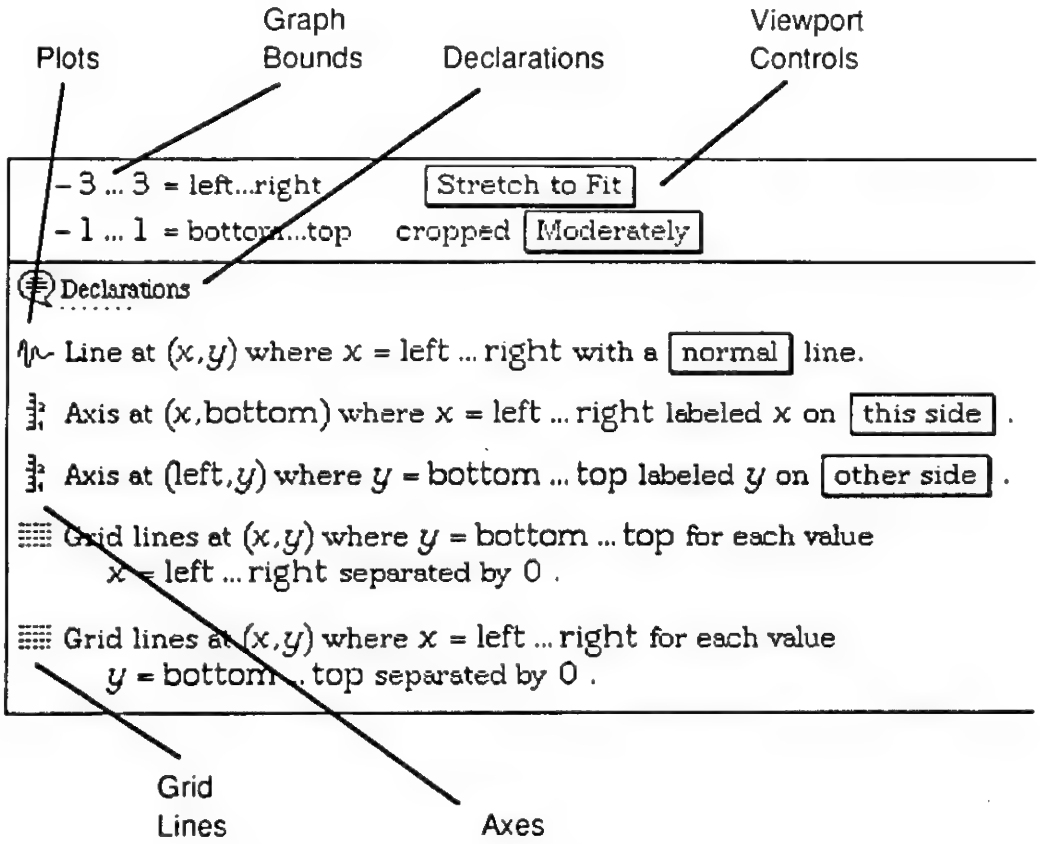
Click on the details box to open or close the graph details.

The Size Box

Click and drag on the size box to change the size and shape of the viewport. Larger graphs consume more memory; more pixels are required to record and display the image. Graph theories with larger viewports take slightly longer to display than smaller theories with the same image. As you drag the size box, the new dimensions are displayed. The size of the viewport is not reflected in the graph details.

Graph Details

The details box allows you to fine tune (and significantly alter) the appearance of a graph theory. The graph details contain information about all graphical elements in a graph theory and how they are displayed in the graph viewport.



Graph Bounds

The graph bounds determine the logical height and width of the graphing area. (The actual screen area of a two-dimensional graph theory is approximately two inches by four inches, and can be adjusted using the size box.) By default, the actual graphing area is somewhat smaller than the viewport. Edit the boundary values as you would any other expression. Set the values to different numbers or to names or expressions that evaluate to numerical values.

The values displayed in the graph bounds change when you use the open-hand pointer or any of the editing icons (except when you change the resolution). For example, zooming out using the Rocket icon is the same as increasing the boundary values. (Strictly speaking, one click

of the Rocket icon is equivalent to doubling the difference between each pair of boundary values.) Similarly, selecting part of a graph with the Knife icon also changes the boundary values.

Viewport Controls

To the right of the boundary values are two pop-up menus. The first determines the *aspect ratio* and the second the degree to which the display is *cropped*.

The aspect ratio is either "True Proportions" or "Stretch to Fit." True Proportions indicates that both axes display the same unit measure. Stretch to Fit adjusts the scale of the axes to fit them into the height and width of the viewport. The default display is Stretch to Fit, unless True Proportions is more appropriate for the values derived from generating the graph. Polar graphs always default to True Proportions.

The more tightly cropped the display the closer the graph bounds are to the edges of the viewport. The default is Moderately cropped. The possible settings are: Very Tightly, Tightly, Moderately Tightly, Moderately, Moderately Wide, Wide, and Very Wide. If the display is Very Tightly cropped, the graph bounds are at the same location as the viewport bounding box.

Declarations

Declarations is a comment proposition (closed by default) that contains various definitions used to construct the graph including the constants bottom, top, left, and right. These names define the bounds of the graphing area in the viewport. Most plots use these defined names. The following names are created by default for a Linear graph:

Declarations

A [Constant] named left behaves as [Horizontal Minimum].

A [Constant] named right behaves as [Horizontal Maximum].

A [Constant] named bottom behaves as [Vertical Minimum].

A [Constant] named top behaves as [Vertical Maximum].

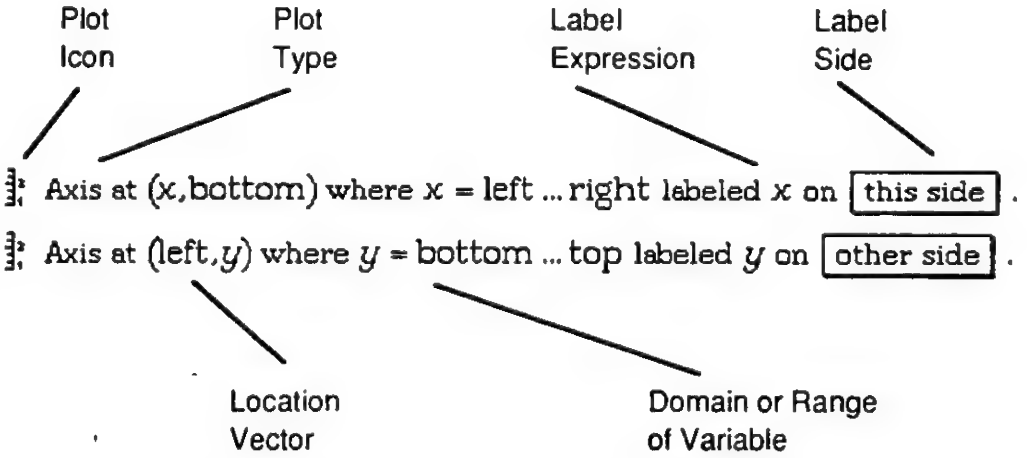
Plots

Each plot (and every other graphical item) is displayed with an icon followed by a sentence describing the plot's attributes. These sentences contain pop-up menus and expressions that you can use to change the plot. For example, Line plots can be normal, heavy, dotted, or dashed; Surface plots can be Transparent, Translucent, Opaque, or

Illuminated. Line and Contour plots are described in detail later in this chapter.

Axes

Each Axis is displayed with an icon and a sentence describing its placement and labeling. Axes (and Grid Lines) are actually simple plots. They calculate the location of a series of points and generate a line connecting those points. This set of points is defined by a location vector. All other plots work on the same principle. By default, each two-dimensional graph theory contains a pair of axes:



The location vector for a default axis consists of one of the equation variables, and a constant defined for that particular graph theory. The value of the constants "bottom" and "left" (which are assigned values in the graph bounds) are independent of the equation variables. In this example, the variable *x* takes on a series of values (between the graph bounds: left ... right). Each of these values is paired with the constant value for bottom. In this way a line is generated along the lower edge of the graphing area. Another line is generated on the far left of this area by the second Axis plot proposition.

Axes are generated with a set of tick marks that display a set of values of the label expression. The label expression is displayed on one side or the other of the axis between the first and second tick marks. By default, labels are taken from the equation. You can change this name to a simple expression containing the named variable (e.g., $3x$, $x/2$). If you want to display a different name (e.g., Speed), either delete the graph, change the variable in the original equation, and generate a new graph, or open the declarations comment, find the name declaration for that variable, and change the spelling of the name there.

You can move axes from one side of the graph to the other (by changing the location vector), add axes to the right or top of the graphing area, and even change the location vector to be dependent on an equation.

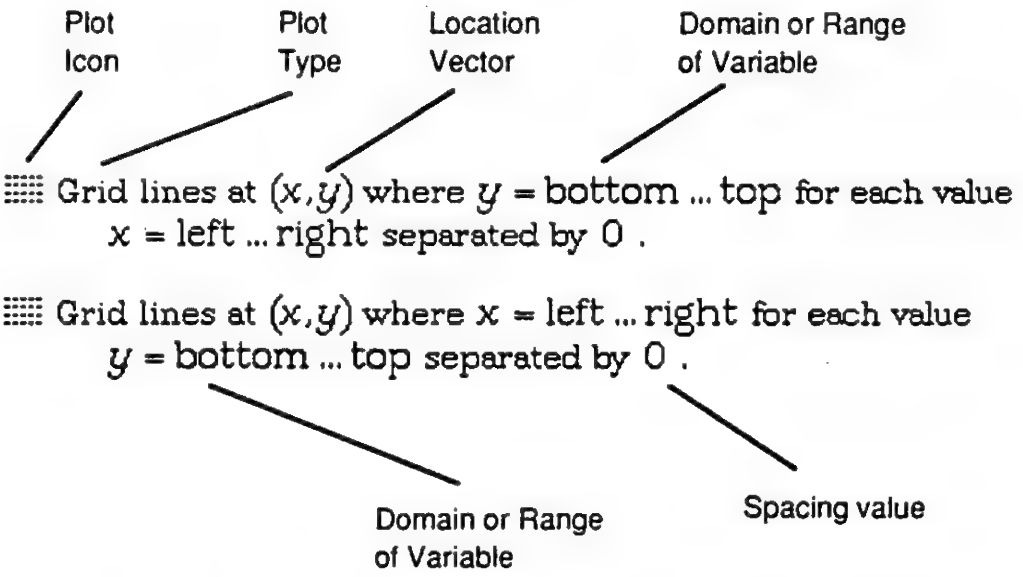
To add a pair of Axes to a graph, select a graph (if there is more than one in the current notebook), and choose **Add Axes** from the **Graphs** menu. A duplicate set of Axes are generated and added to the graph details. To make a box of axes, change the two constants bottom and left to top and right:

Axis at (x, top) where x = left ... right labeled x on this side .

Axis at (right, y) where y = bottom ... top labeled y on other side .

Grid Lines

Each set of (usually parallel) Grid lines is displayed with an icon and a sentence describing its placement. Grid Lines are also simple plot propositions. By default, each two-dimensional graph theory contains two Grid line plot propositions:



As with Axes propositions, Grid lines utilize a location vector to determine where the lines are drawn. The first proposition of this pair creates a set of vertical lines; the second creates the horizontal lines.

The spacing value determines the distance between Grid lines. If the spacing value is zero (the default), one grid line is drawn to match each tick mark on the axis (if you are using default Axes). To draw a grid line

at every unit, change this value to one. For lines every half unit, set the spacing value to 0.5.

On color monitors, Grid lines appear as light purple. On black and white monitors (and when printed on PostScript printers), they are drawn as dotted lines. Grid lines for major increments are thicker or darker than other grid lines. Major increments are at zero and at other large, round numbers. Zero always is a major increment.

Creating 2D Graphs

There are seven types of graphs that, when newly created, generate a two-dimensional graph theory. These graphs are:

Graph Type	Menu Option
Parametric	$x = f(t), y = g(t)$
Linear	$y = f(x)$
SemiLog	$y = f(x)$
Polar	$y = f(x)$
Density	$z = f(x, y)$
Contour	$z = f(x, y)$
Zero Contour	$z = f(x, y)$

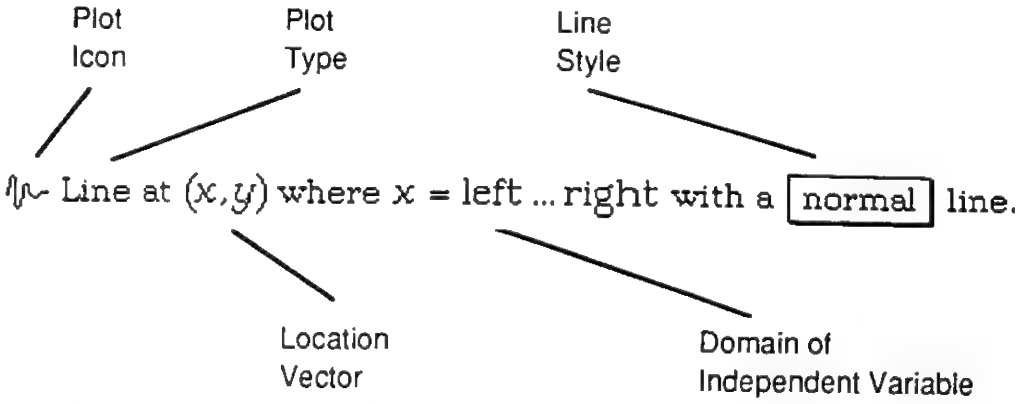
Parametric, Linear, SemiLog and Polar graphs all generate Line plots from the selected function and are drawn with a Normal line. This line can be adjusted in the pop-up menu of the proposition to be normal, heavy, dotted, or dashed. Density graphs generate Surface plots, and both types of contour graphs generate Contour plots.

This section describes Line and Contour plot propositions and then gives examples of each type of graph created in two dimensions. Surface plots are described in the following section.

Default bounds of –3 and 3 are often used when a two-dimensional graph is first generated. (In many cases other default bounds are generated.) These bounds set the domain of the independent variable(s). Before creating a graph, you can explicitly set one or both boundary values with an inequality (e.g., $x > 0$, $x \leq 100$). If only one bound is set, the other bound is established by the program. You can also edit these bounds in the details box once the graph is created.

Line Plots

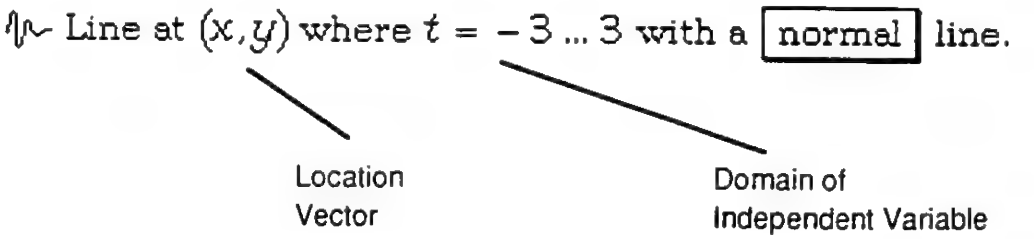
Line plot propositions contain the following information:



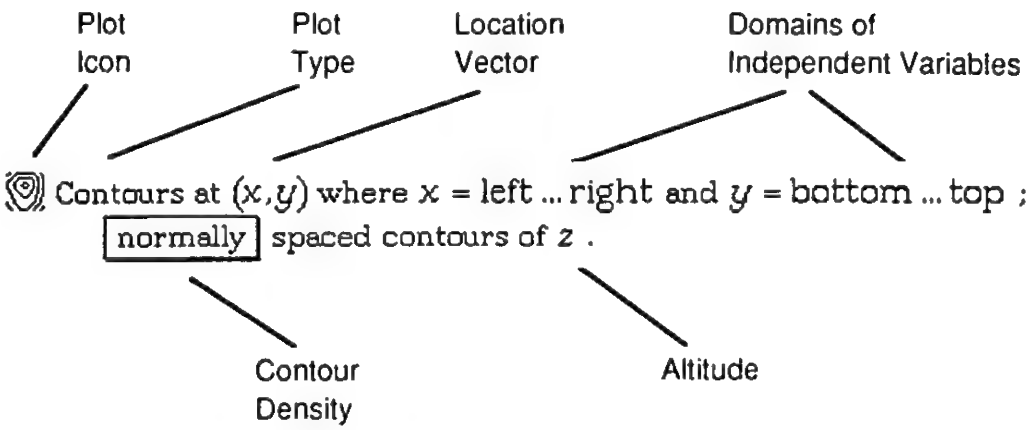
The location vector and the domain of the independent variable determine the point-by-point location of the the plot. The first value in the location vector determines the horizontal (left ... right) location of each point, and the second value determines the vertical (bottom ... top) location. In this example, the boundary values (as set in the first line of the graph details) are used to generate a series of *x* values. The *y* values for each point are generated by plugging successive *x* values into the original equation.

You can replace the location vector with any expression that evaluates to a two-element vector with real values.

Parametric plots work slightly differently in that the domain is set to specific numerical values not affected by the graph's boundary values:



Contour plots contain the following information:



Contour plots work somewhat like Line plots. The first value in the location vector determines the horizontal location of a point and the second determines the vertical location, and each domain specifies a series of values to use for each. A standard Contour plot generates several lines as determined by the original equation, the density pop-up menu and the altitude value. Each contour is generated by plugging successive values of the location vector into the original equation.

The altitude value (in this case z), can be any expression dependent on one or both of the domain variables. If the expression is not dependent on these values, no plot lines are drawn. With the pop-up menu set to "normally," between eight and twenty lines are generated. The other menu settings halve or double the number of contour lines drawn. The menu choices are: at zero only (which specifies a single line), very sparsely,sparsely, normally, densely, very densely.

The spacing of contours is always some power of ten times two, five, or ten. For example, contours may be placed placed every 0.02 units, 0.05 units, 0.1 units, 0.2 units, or 0.5 units, and so on.

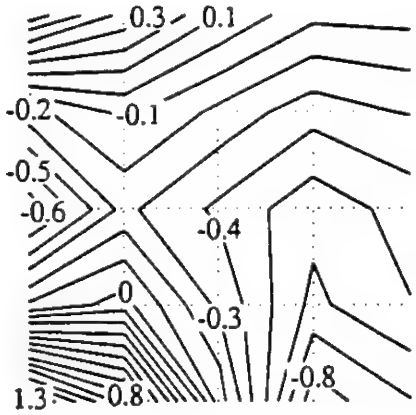
For predictable contour spacing, you must explicitly set a range of values for the altitude. Edit the altitude expression so it reads, z = minimum ... maximum. This is essential if you intend to animate a graph with a Contour plot. If the range is not specified, the program may choose different contour spacings for each rendering. If the altitude value is above or below the specified range, a few contour lines are drawn, but most are not.

Because the ratio between two and five is greater than two, choosing the next menu setting may *not* change the number of contour lines. For example, if the range of altitude values is $z = 0 \ldots 1$, the two settings "sparsely" and "very sparsely," draw the same number of lines.

Contour lines are labeled with their altitude values. No more than one label is generated for each altitude value. If there are multiple hills in different parts of a Contour plot, and each hill passes through the same altitude contours, no more than one of each contour is labeled. Some altitude values are not labeled at all if the labels are too close to each other. For these reasons, there may be contours whose altitude values are not immediately apparent.

Contour plots are drawn in panels, like Surface plots. Each panel contains a set of straight lines that connect with other lines from neighboring panels at the edges. The resolution icons determine the number of panels, as with Surface plots. The resolution value indicates the number of panels in each direction. Therefore, the default display of a Contour plot generated from a particular equation is not as smooth as a Line plot generated from a similar equation, even though the resolution values are the same.

The following Contour plot has a resolution value of four (four panels on each side for a total of sixteen panels). A set of Grid lines has been added to the graph delimiting each panel. Within each panel the lines of the plot are straight.



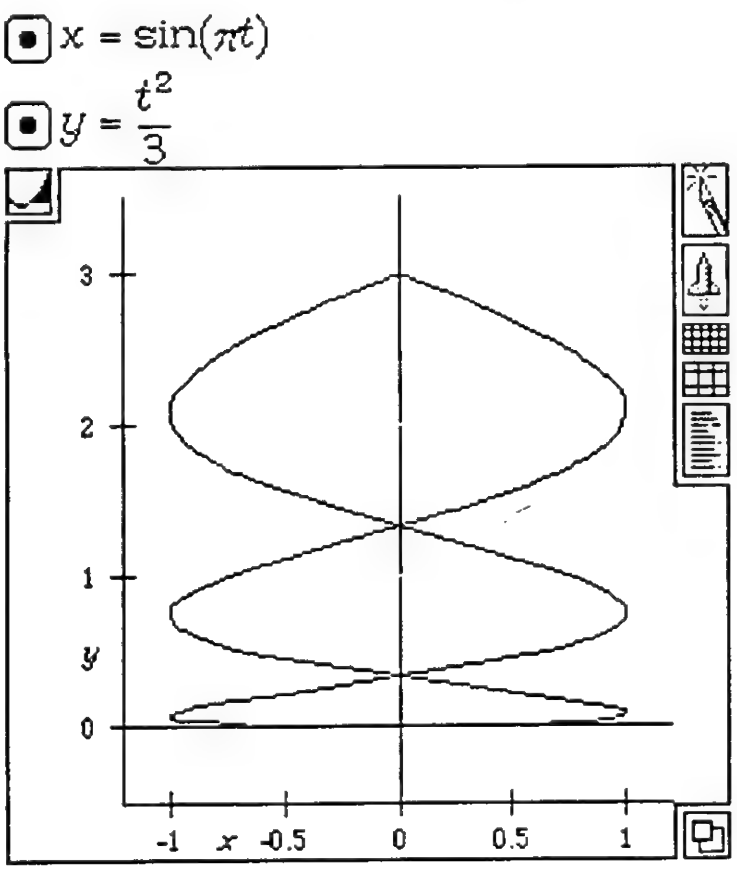
2D Graphs

This section describes each type of graph that generates a two-dimensional graph theory. To create any one of these graphs, choose a command from the Graphs menu. There are seven two-dimensional graphs:

- Parametric
- Linear
- SemiLog
- Polar
- Density
- Contour
- Zero Contour

Parametric

To create a Parametric graph, select two equations of the form $x = f(t)$, $y = g(t)$ and choose the first option on the Graphs menu:



The description of this plot is shown in the graph details:

$\sim$ Line at (x, y) where $t = -3 \ldots 3$ with a normal line.

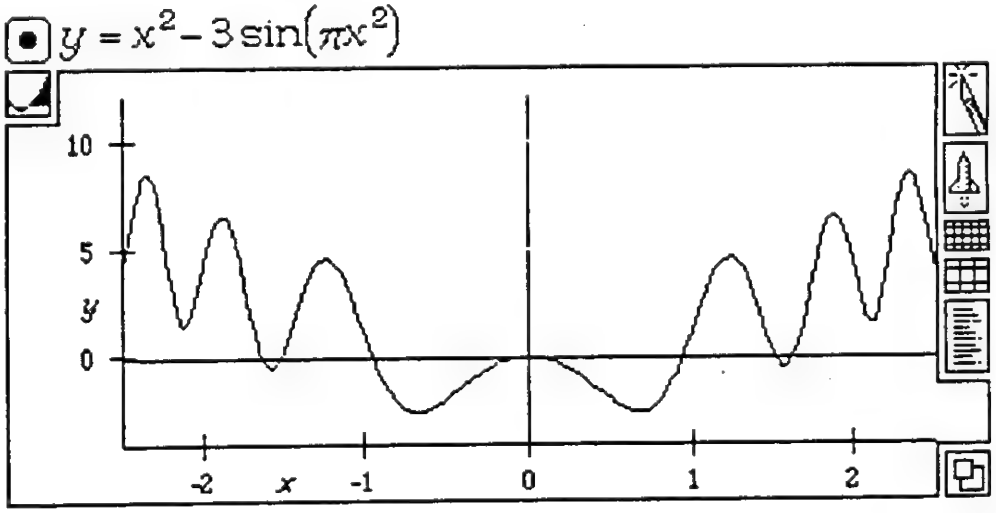
In this graph, x and y are both dependent variables; t is the independent variable. No boundary values were explicitly indicated for t; defaults of –3 and 3 were used to generate the graph.

The dependent variable in the first selected equation becomes the first variable in the location vector and is displayed along the horizontal axis.

Parametric graphs are displayed after the second selected equation.

Linear

To create a Linear graph, select an equation of the form y = f(x) and choose the first sub-option of the second option on the Graphs menu (or press Command-G):



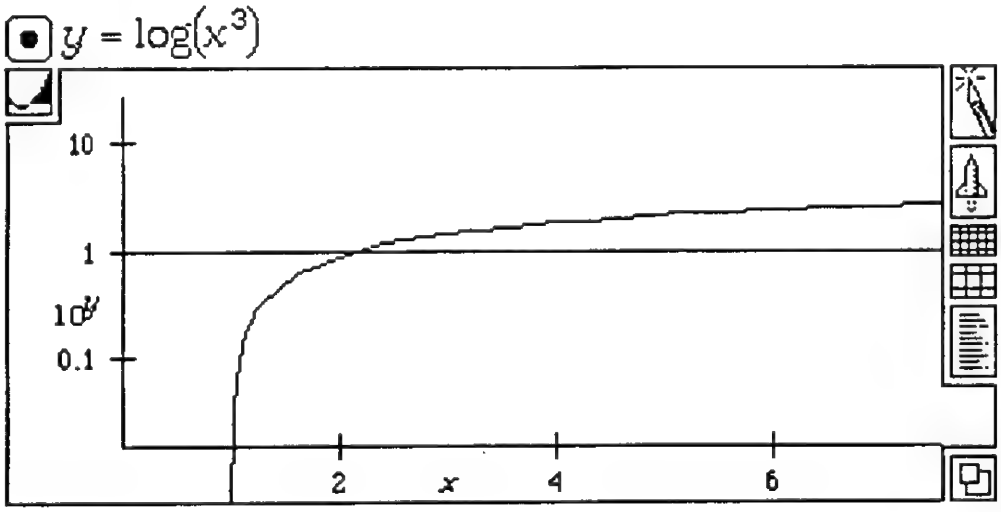
The description of this plot is shown in the graph details:

$\sim$ Line at (x, y) where x = left ... right with a normal line.

The bounds of this graph were edited to be –2.3 ... 2.5 = left ... right and –4 ... 12 = bottom ... top.

SemiLog

To create a SemiLog graph, select an equation of the form y = f(x) and choose the second sub-option of the second option on the Graphs menu:

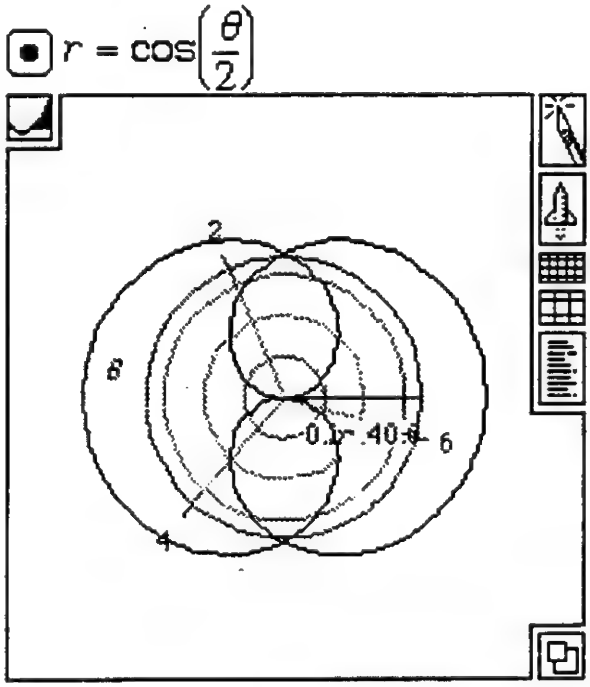


The description of this plot is shown in the graph details:

Line at $(x, \log[y])$ where x = left ... right with a [normal] line.

Polar

To create a Polar graph, select an equation of the form y = f(x) and choose the third sub-option of the second option on the Graphs menu:



The description of this plot is shown in the graph details:

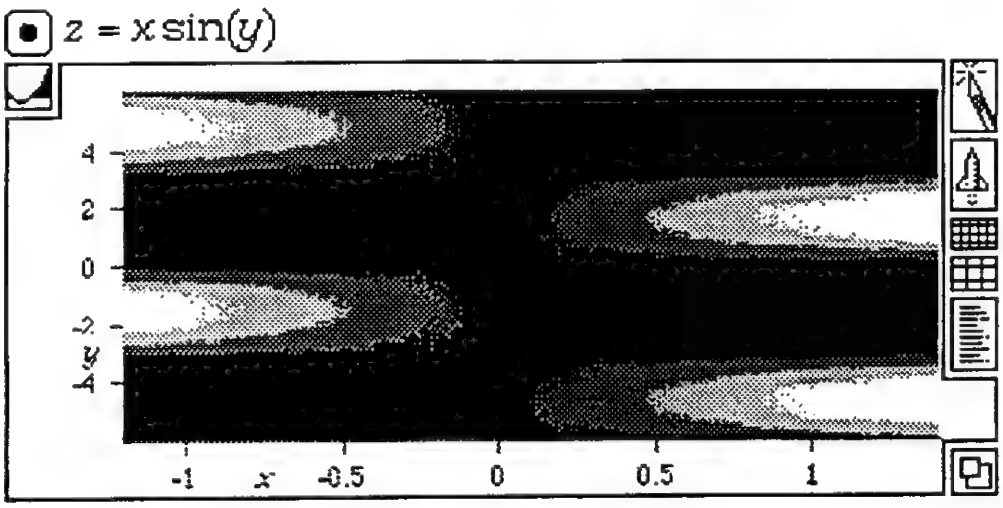
Line at FromPolar(r, θ) where θ = 0 ... 4π with a [normal] line.

For this graph, the domain of the independent variable (θ) was increased from the default of 0 ... 2π to 0 ... 4π.

Polar plots use the predefined function FromPolar to translate data-point values from polar to rectangular coordinates. The first time you create a polar graph, a dialog box asks you to confirm the inclusion of this function in the current notebook. The name declaration for FromPolar is placed in the Declarations comment at the top of the current notebook and is used by all subsequent Polar graphs. See the Expressions chapter for more information about predefined names and behaviors.

Density

To create a Density graph, select an equation of the form z = f(x, y) and choose the first sub-option of the third option on the Graphs menu:



The description of this plot is shown in the graph details:

Surface at (x,y) where x = left ... right and y = bottom ... top ;
Opaque surface has no mesh and is shaded using Gradient coloring;
z selects a color from Black ... White .

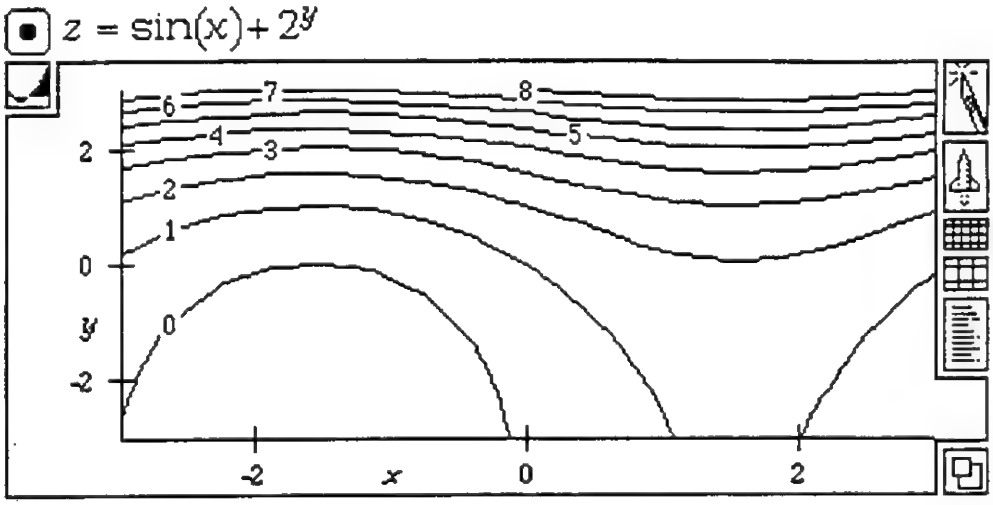
A density graph consists of a two-dimensional graph theory, a Surface plot, and a pair of Axes. Surface plots are usually displayed in three-dimensional graph theories; density plots are an exception. The plot is displayed with an opaque surface, with no mesh. The z values are displayed as a range of grays from black-to-white. All pop-up menus can be adjusted.

For this graph, the resolution was increased from the default of eight to the maximum of 128 by clicking on the higher-resolution icon.

See the section on three-dimensional graphs, later in this chapter, for more details about Surface plots.

Contour

To create a Contour graph, select an equation of the form z = f(x, y) and choose the second sub-option of the third option on the Graphs menu:



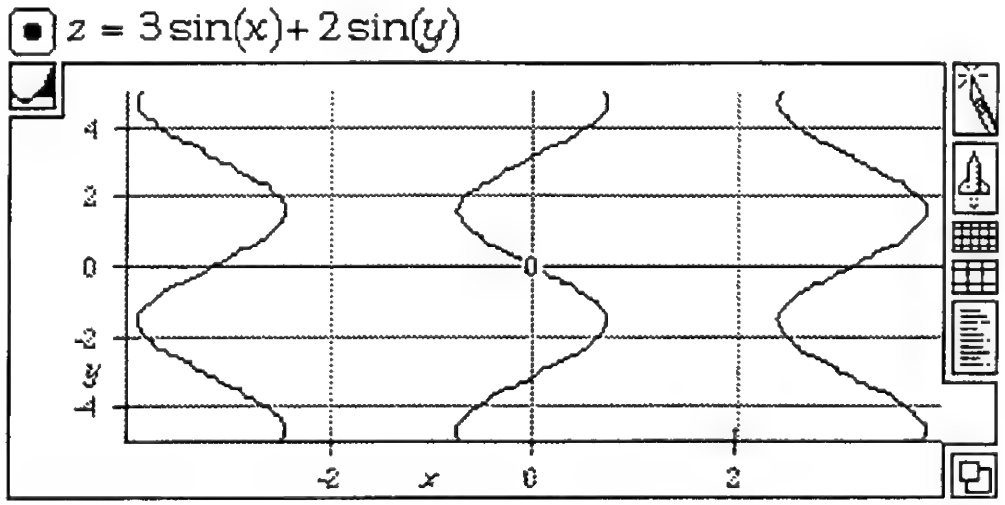
The description of this plot is shown in the graph details:

Contours at (x,y) where x = left ... right and y = bottom ... top ;
normally spaced contours of z .

Contour plots display the relationship between three variables in two dimensions. The dependent variable (z) is displayed as contour lines of equal value as on a topographical map. Using the pop-up menu, contours can be placed at zero only, very sparsely, sparsely, normally (the default), densely, or very densely. To create a graph that displays a single zero contour initially, use the next option.

Zero Contour

To create a Zero Contour graph, select an equation of the form z = f(x, y) and choose the second sub-option of the third option on the Graphs menu:



The description of this plot is shown in the graph details:

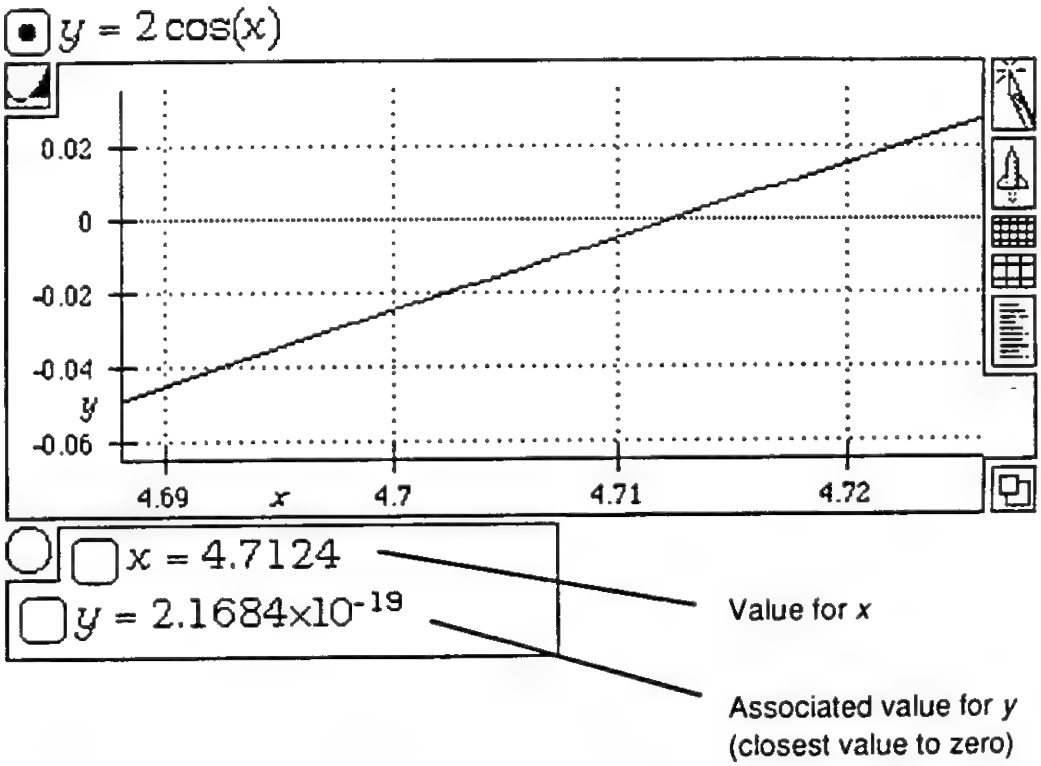
Contours at (x,y) where x = left ... right and y = bottom ... top ;
at zero only spaced contours of z .

Zero Contour plots display the single contour line where z equals zero. Contour density can be adjusted with the pop-up menu, as with a normal Contour plot, to very sparsely, sparsely, normally, densely, or very densely spaced. Zero Contour graphs display a set of Grid lines, whereas normal Contour graphs do not. Other than this, the graph is no different than a normal Contour graph.

For this graph, the resolution was increased from the default of eight to 64, and the graph bounds were changed to –4 ... 4 = left ... right, and –5 ... 5 = bottom ... top.

Finding Roots

To find the root of a Line plot displayed in a two-dimensional graph theory, zoom in (with the Rocket or the Knife) on a portion of the graph that contains only one crossing of the horizontal (x) axis. Then select **Find Root** from the **Graphs** menu. A new case theory appears immediately following the graph theory displaying values for the two variables:



If there is more than one root (value for x) visible in the viewport, one is selected arbitrarily, or you receive an error message.

You can evaluate several roots by focusing on different portions of the graph in succession. You generate a new case theory each time you choose Find Root.

You can also use Find Root for simultaneous numerical equation solving. By creating two zero Contour plots, and finding a crossing of the two plots, you can solve two equations in two unknowns. For example, consider finding solutions for the two equations:

$$\frac{\log(y)}{x} = 2y\sin(x)$$

$$e^x y = 2x^2$$

Use the Move Over manipulation to create two equations that equal zero. Then rename the zeros zero_1 and zero_2:

$$\square\ \frac{\log(y)}{x} = 2y\sin(x)$$

$$\triangle\ 0 = -\frac{\log(y)}{x} + 2y\sin(x)$$

$$\square\ \text{zero}_1 = -\frac{\log(y)}{x} + 2y\sin(x)$$

$$\square\ e^4 + e^x y = 2x^2$$

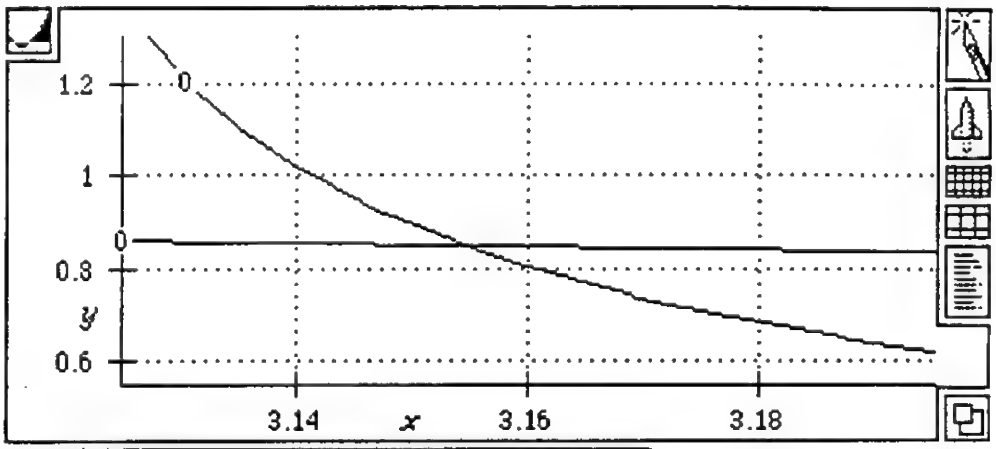
$$\triangle\ 0 = 2x^2 - e^x y - e^4$$

$$\square\ \text{zero}_2 = 2x^2 - e^x y - e^4$$

Create a zero Contour graph from the first equation. Declare zero_1 to be the dependent variable, and x and y to be the independent variables. Then add a second Contour plot to the graph theory based on the second equation. Use the **Add Contour Plot** option on the **Graphs** menu. Declare zero_2 to be the dependent variable, and x and y to be the independent variables. Open the graph details and adjust the pop-up menu of the second Contour plot to "at zero only" (the first Contour plot was created with this setting):

Contours at (x, y) where x = left ... right and y = bottom ... top ;
[at zero only] spaced contours of zero_2 .

Then use the Knife, the open-hand pointer, and the Rocket to close in on a point where the two plots cross. When a single crossing point is in view, choose **Find Root**. A case theory is generated with values for x and y and associated values (close to zero) for $zero_1$ and $zero_2$:



$x = 3.1549$
$y = 0.84889$
$zero_2 = 0$
$zero_1 = 1.3383 \times 10^{-19}$

Three-Dimensional Graphs

Three-dimensional graphs consist of a three-dimensional graph theory (the shell) and one or more two- or three-dimensional plots (Line, Surface, or Contour) and associated Axes and Grid lines. This section presents:

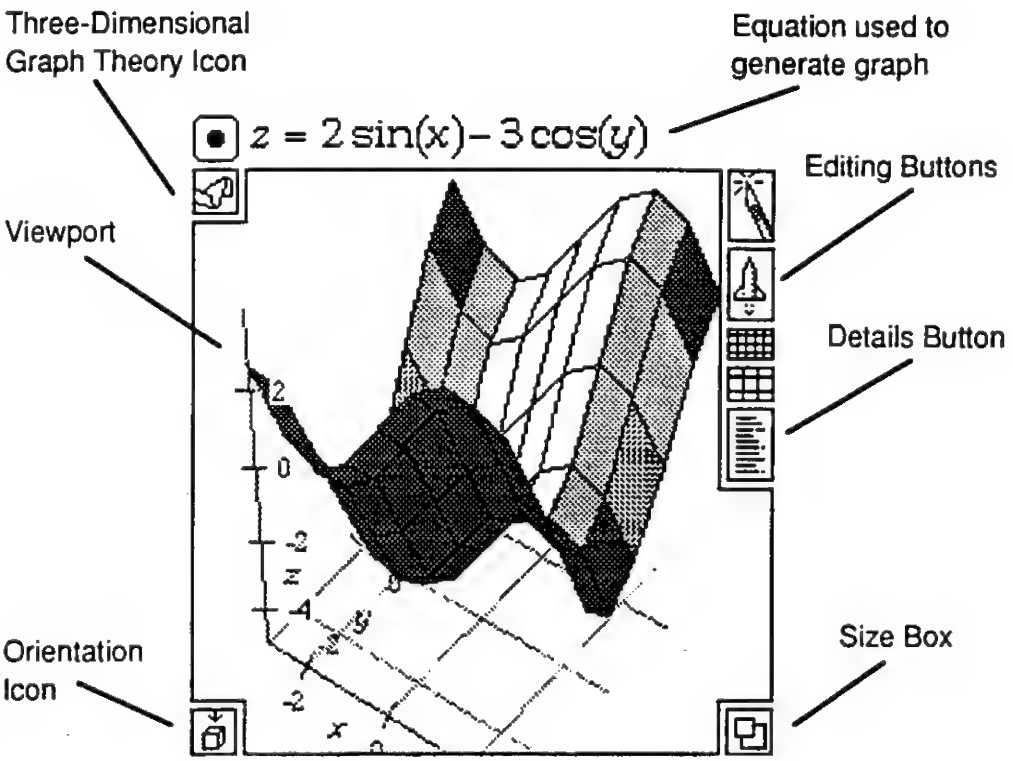
- Icons for editing three-dimensional graph theories
- Graph details of three-dimensional graph theories
- A list of graphs that generate three-dimensional graph theories
- A description of each type of Surface plot proposition

Once created, plot propositions can be adjusted, edited, and moved around like any other proposition. To move a graph element, select a plot and Cut (or Copy) and Paste it into another graph theory. Plot propositions are meaningless outside of a graph theory. Axes and Grid lines are also plots, and can be moved and edited in the same way.

You can move Surface plots between three-dimensional graph theories without difficulty. However, it's often easier to select an equation and choose the appropriate menu option on the Graphs menu to add Line, or Contour plots, Axes or Grid lines, or even additional Surface plots.

To add a plot (or axis or set of grid lines) to a graph theory, select an equation (and a graph theory if there is more than one), and choose **Add Line Plot**, **Add Surface Plot**, or **Add Contour Plot** (or **Add Axes** or **Add Grid Lines**) from the **Graphs** menu.

Most three-dimensional graph controls are similar to the two-dimensional graph controls. However, the open-hand pointer works quite differently, and an extra icon, the "orientation icon," is displayed in the lower left-hand corner of the graph theory.



To *rotate* (rather than slide) a three-dimensional graph, click in the viewport with the open-hand pointer, and drag the image left, right, up, or down. The portion of three-dimensional space displayed in the viewport rotates as if you had your hand on a globe. As you drag, a cube or rectangular box shows the bounds of the graph. When you release the mouse, the graph redraws with the new orientation. Unlike clicking and dragging in a two-dimensional graph theory, the graph's boundary values do not change.

The outline cube indicates every 15 degrees of rotation (in latitude and longitude). The line width increases at each multiple of 15 degrees to two pixels. If you rotate the cube by multiples of 15 degrees longitude *and* 15 degrees latitude, the line density increases to four pixels.

Rotational freedom is limited to rotating the display around its vertical axis, and rotating the vertical axis towards and away from you. Hold down the Option key to tilt the vertical axis and rotate the graph more freely. In this mode, you can rotate the display in the line of sight by dragging in a wide circle. Do this in a direction opposite to the direction you want to rotate the model. (This technique works on

the same principal as rotating a tennis ball on a table top with the palm of your hand. The direction of rotation is the opposite of the effected rotation.) Click on the Orientation icon to return the display to its original orientation.

To change the graph bounds of a three-dimensional graph, use the Knife or Rocket icons as you would for a two-dimensional graph, or edit the boundary values in the graph details.

The Knife

Click on the Knife icon to *pick up* the Knife (the cursor becomes a knife). Click and drag across a selection of surface panels to look at just that part of a plot. The portion of the graph you select can be most of the surface or just a tiny portion. It's easiest to use the knife if your view is of the top (or bottom) of a plot, or perpendicular to the portion of the surface you want to select. Using the Knife changes the values of the graph bounds.

The Rocket

Click on the Rocket icon to *zoom out* (from the center of the viewport) by a factor of two. Press Option and click on the Rocket to *zoom in* by a factor of two. After a moment, the graph redraws. To zoom in or out by more than a factor of two, click the icon repeatedly. To zoom out by a factor of 1000, click on the icon ten times. Zooming in and out is reflected in the values of the graph bounds.

The Resolution Icons

Click on the dense grid to increase the data points used to create the plots in the graph by (approximately) a factor of two. Click on the open grid to (approximately) halve the number of data points. The more data points used, the smoother the plot surface(s), and the longer it takes to generate the image.

When you click on a resolution icon, the current value and the destination value appear briefly to the left of the icon (e.g., 8 -> 16). Resolution values are 1, 2, 4, 8, 16, 32, 64, 128 (default: 8). For Surface Plots and Contour Plots, the value indicates the number of panels on

each side of the displayed plot. For Line Plots, Axes, and Grid Lines, the value indicates the approximate number of line segments used to create a 90 degree curve. Plot resolution is not reflected in the graph details.

The Details Box

Click on the details box to open or close the graph details.

The Size Box

Click and drag on the size box to change the shape of the viewport. Larger graphs require more memory (more pixels), and take slightly longer to generate. Changing the size of the viewport is not reflected in the graph details.

The Orientation Icon

Click on the orientation icon at any time to return to the original orientation and sizing of a display. The orientation icon displays the side of the boundary volume shown in the viewport:

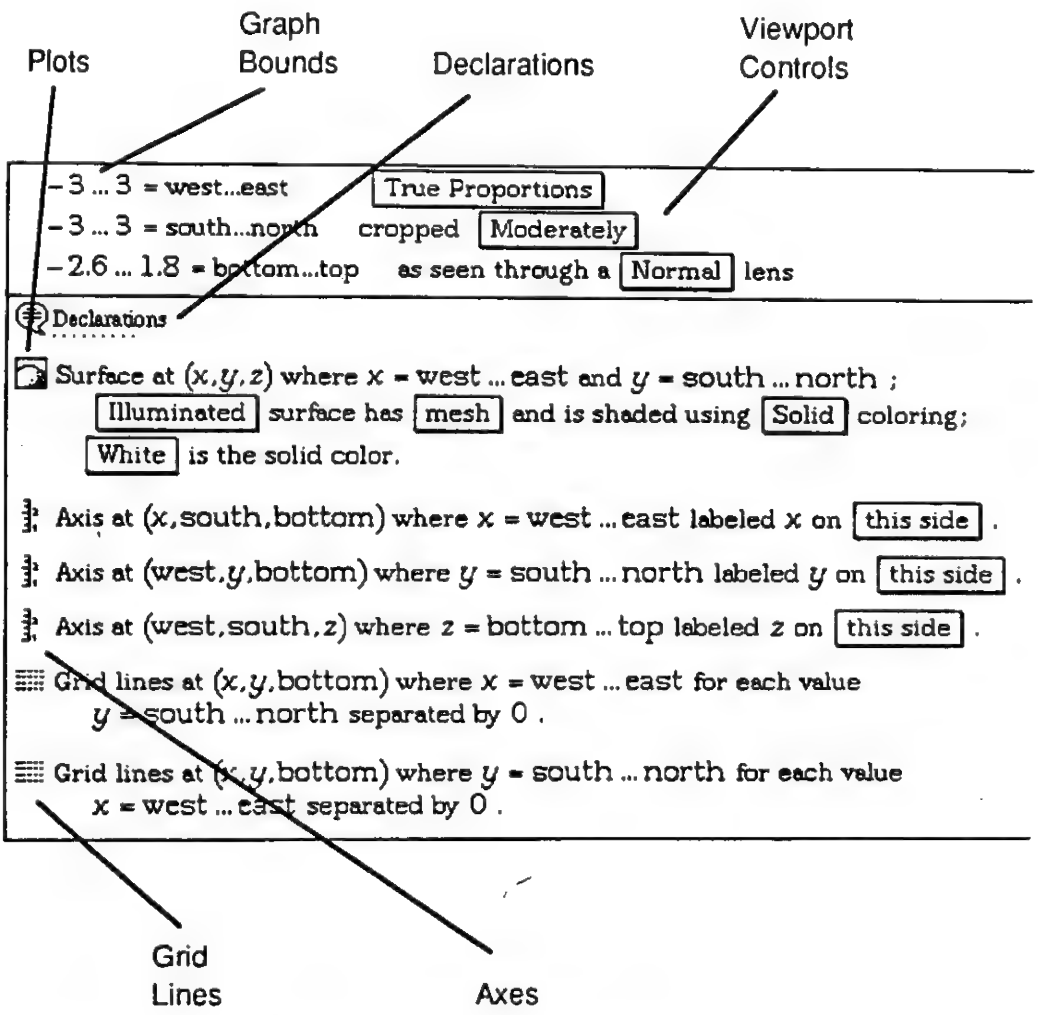
front back

right top

left bottom

Graph Details

The details box allows you to fine tune (and significantly alter) the appearance of a graph theory. The graph details contains information about all graphical elements in a graph theory, and how they are displayed in the graph viewport:



Graph Bounds

The graph bounds determine the logical height, width, and depth of the graphing area. Edit the boundary values as you would any other expression. Set the values to different numbers or to names or expressions that evaluate to numerical values. Using the Rocket or Knife icon deletes the current boundary values and replaces them with new values. Rotating a three-dimensional graph does not change the boundary values and is not reflected in the graph details.

Viewport Controls

To the right of the boundary values are three pop-up menus. The first determines the *aspect ratio*, the second the degree to which the display is *cropped*, and the third the type of *lens* used to view the graph.

The aspect ratio is either "True Proportions" or "Stretch to Fit." True Proportions indicates that all three axes display the same unit measure. Stretch to Fit adjusts the scale of the axes to fit an exact cube. The default display is Stretch to Fit, unless the values derived from generating the graph are close to True Proportions. Spherical 3D, Cylindrical 3D, and Complex 3D graphs are always created using True Proportions.

To adjust how tightly a graph is displayed, choose an item from the cropping pop-up menu. A tightly cropped view displays a tight close up of a three-dimensional graph, whereas a loosely cropped view includes the entire graph, all axes labels, and a great deal of white space. The default is Moderately cropped (a small portion of the graph is usually outside the viewport). The possible settings are: Very Tightly, Tightly, Moderately Tightly, Moderately, Moderately Wide, Wide, and Very Wide.

The lens setting specifies the amount of foreshortening or perspective. A Wide Angle lens exaggerates the three-dimensional effect. The Telephoto setting reduces this effect, and Infinitely Distant creates an isometric projection (no foreshortening). The possible settings are: Very Wide Angle, Wide Angle, Slight Wide Angle, Normal, Slight Telephoto, Telephoto, and Infinitely Distant.

Declarations

Declarations is a comment proposition (shown closed by default) that contains various definitions used to construct the graph including the constants bottom, top, west, east, south, and north and possibly other declarations depending on the type of graph (e.g., the constant radius for Spherical 3D graphs, the variables real and imaginary for Complex 3D graphs). The following name declarations are created by default for a Color 3D graph:

Declarations

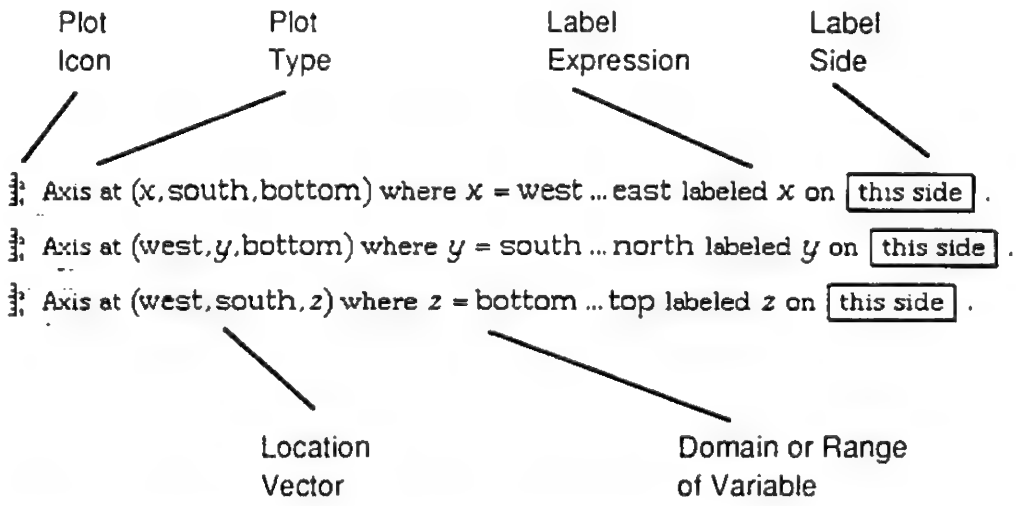
- A [Constant] named west behaves as [3D X-Minimum].
- A [Constant] named east behaves as [3D X-Maximum].
- A [Constant] named south behaves as [3D Y-Minimum].
- A [Constant] named north behaves as [3D Y-Maximum].
- A [Constant] named bottom behaves as [Vertical Minimum].
- A [Constant] named top behaves as [Vertical Maximum].

Plots

Each plot (and every other graphical item) is displayed with an icon followed by a sentence describing the attributes of the plot. These sentences contain pop-up menus that you can use to change the plot's attributes. For example, Line plots can be Normal, Heavy, Dotted, or Dashed; Surface plots can be Transparent, Translucent, Opaque, or Illuminated.

Axes

Each Axis is displayed with an icon and a sentence describing its placement and labeling. Axes are simple plot propositions that use certain constants to generate a frame of reference for the section of three space shown in the viewport. They calculate the location of a series of points and generate a line connecting those points. This set of points is defined by a location vector. By default, each three-dimensional graph contains a trio of Axes:



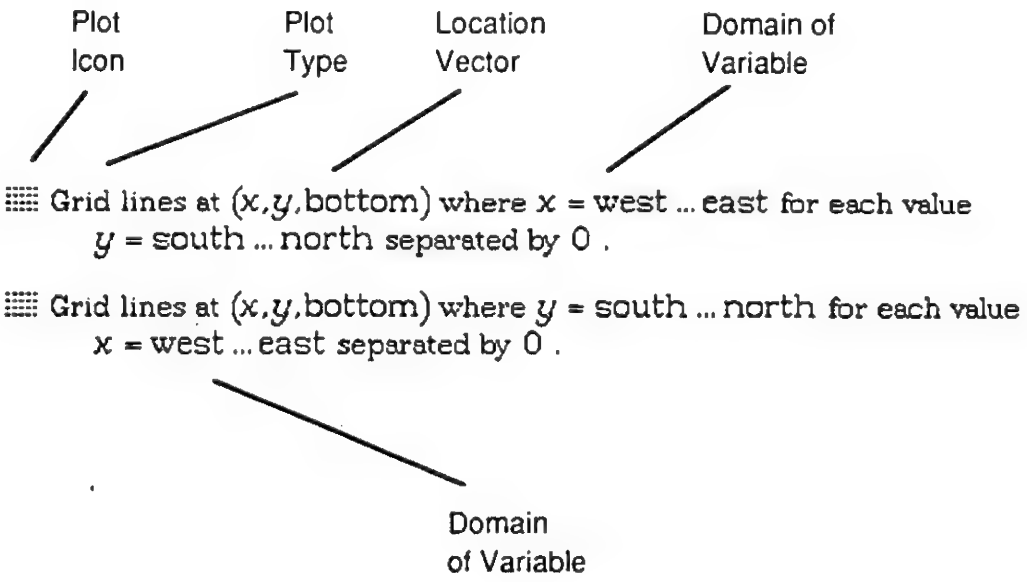
Axes for three-dimensional graphs are generated the same way two-dimensional axes are generated. Two constants are grouped with one of the equation variables in a location vector to generate a series of points that form a straight line. Spherical graphs generate two curved axes and one straight axis, whereas Cylindrical graphs generate one curved axis and a pair of straight ones.

Axes are generated with a set of tick marks that display a set of values of the equation variable. This expression is displayed on one side or the other of the axis between the first and second tick marks.

You can move Axes from one edge of the graphing area to another (by changing the location vector), add Axes to the other edges of the graphing area, and even change the location vector to be dependent on an equation.

Grid Lines

Each set of (usually parallel) Grid lines is displayed with an icon and a sentence describing its placement. Grid lines are also simple plot propositions. By default, three-dimensional graphs, other than spherical and cylindrical graphs, contain two Grid line plot propositions. Spherical and Cylindrical graphs do not generate Grid lines.



Grid Lines for rectangular three-dimensional graphs are displayed at the bottom of the segment of three space displayed in the viewport. As with axis plot propositions, Grid lines can be moved any where in this space, added, or deleted.

Grid lines utilize a location vector to determine where the lines are drawn. By default, the graph bounds are used to determine the location of the Grid lines. The first proposition of this pair creates a set of lines drawn parallel to the *x* axis (west ... east); the second creates a set drawn parallel to the *y* axis (south ... north).

You can also set the range of *x* values to any expression that evaluates to a range of two real numbers. For example, if you set the range for the first of these plots to *x* =west +1 ... east −1, the proposition generates a set of lines parallel to the *x* axis that start one unit in from each side.

The spacing value determines the distance between Grid lines. If the spacing value is zero (the default), one grid line is drawn to match each

tick mark on the axis (if you are using default Axes). To draw a grid line at every unit, change this value to one. For lines every five units, set the spacing value to 5.

On color monitors, Grid lines appear as light purple. On black and white monitors (and when printed on PostScript printers), Grid lines are drawn as dotted lines. Grid lines for major increments are thicker or darker than other grid lines. Major increments are at zero and at other large, round numbers. Zero is always a major increment.

Creating 3D Graphs

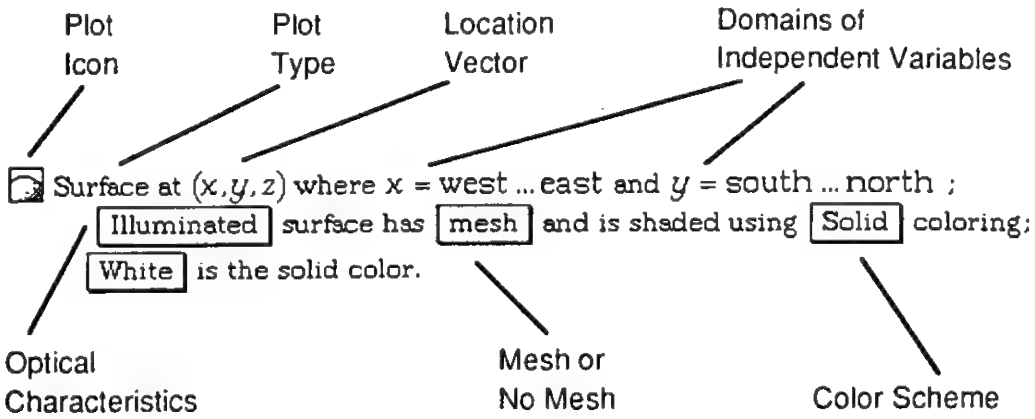
There are five types of plots that, when newly generated, create a three-dimensional graph theory. These are:

Graph Type	Menu Option
Complex 3D	$y = f(x)$
Color 3D	$z = f(x, y)$
Illuminated 3D	$z = f(x, y)$
Spherical 3D	$z = f(x, y)$
Cylindrical 3D	$z = f(x, y)$

Complex 3D, Color 3D, Illuminated 3D, Spherical 3D, and Cylindrical 3D graphs all create Surface plots generated from the selected function.

You can specify the domain of the independent variables before creating a graph (e.g., $x > 0$, $x < 10$), or let the program generate default values. These values are displayed at the top of the details box and can be edited easily after creating a graph.

Surface plot propositions display the following information:



Surface plots are naturally more complex than Line plots.

The first line of a Surface plot proposition contains the location vector and the domains of the independent variables. These are used to generate the vertices of the surface panels. The first value in the location vector determines the west ... east location for each vertex and the second value determines the south ... north location. Value ranges for the domains are set in the first lines of the graph details (the graph bounds). Values for the third element in the location vector (used to establish the bottom ... top location of each vertex) are generated by plugging a series of values for the independent variables into the original equation.

The second line of a Surface plot proposition contains three pop-up menus. These menus specify the optical characteristics of the surface, whether the surface is displayed with a mesh (a rectangular grid connecting the vertices generated from the location vector), and the fundamental color scheme used to display the surface.

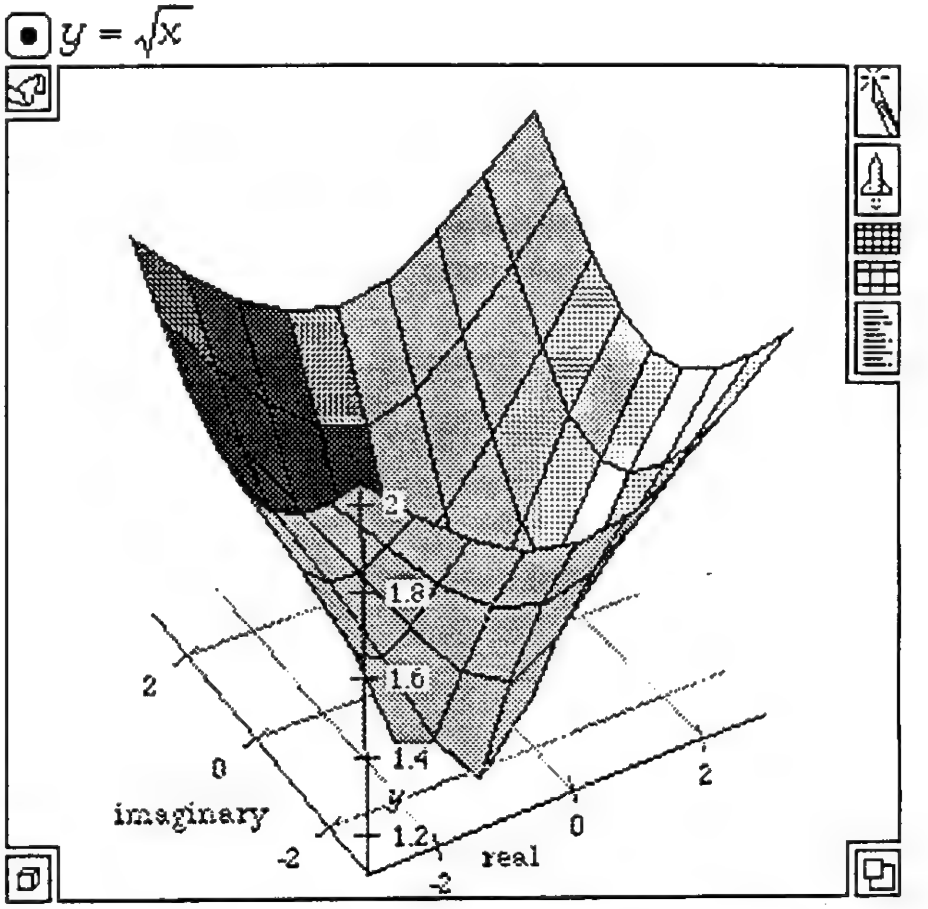
The third line of a Surface plot is determined by the type of color scheme specified in the last menu of the second line. Each color scheme can be adjusted by altering the values in the third line of the plot proposition.

3D Graphs

Each graph initially generates a specific variant of Surface plot as described in the following sections. See the section on Surface plots, later in this chapter, for more information.

Complex 3D

To create a Complex 3D graph, select an equation of the form y = f(x) and choose the fourth sub-option of the second option on the Graphs menu:



The default description of this plot is shown in the graph details:

Surface at (real, imaginary, |y|) where real = west ... east and imaginary = south ... north ;
Opaque surface has mesh and is shaded using Complex coloring;
y determines brightness and hue (from magnitude and phase).

Complex 3D graphs generate a Surface plot specifically designed to display complex numbers, using the color scheme Complex. The location vector consists of the two variables real and imaginary, and the absolute value of the dependent variable, *y*. A Complex 3D graph splits the independent variable into its real and imaginary components and treats each of these as independent variables. The two

variables, real and imaginary, are defined for this graph only and contained in the graph details under the Declarations proposition:

A Variable named real behaves as defined by user .

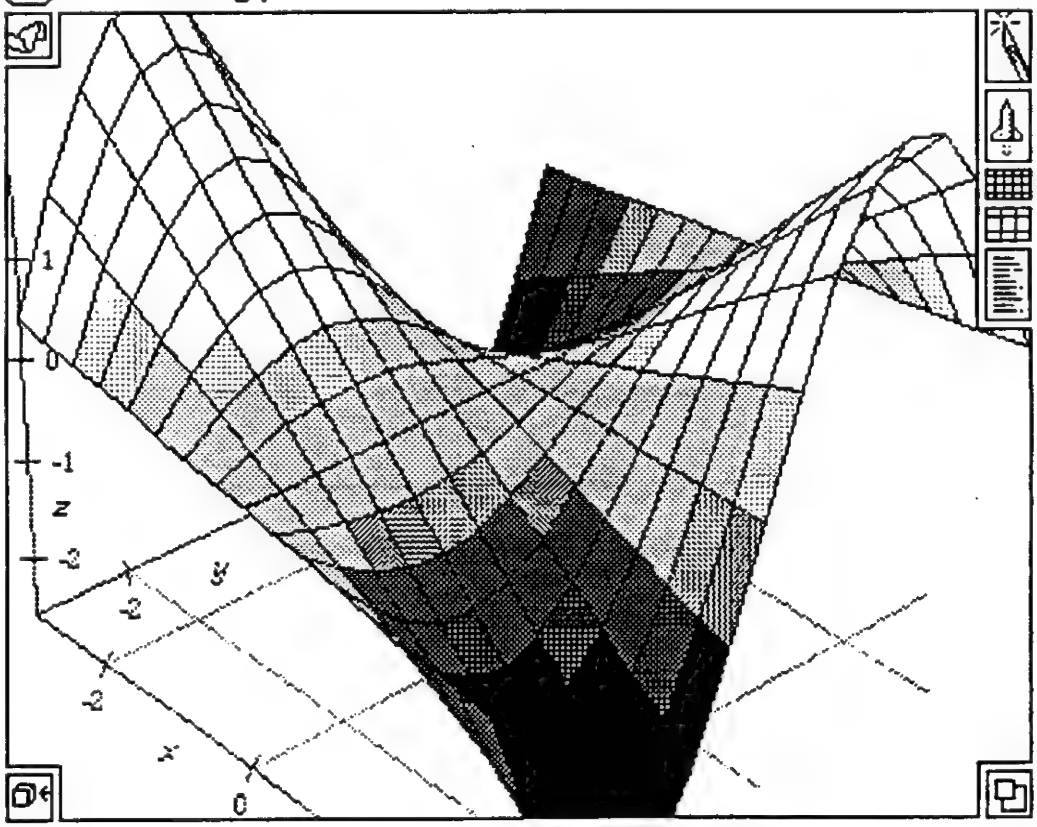
A Variable named imaginary behaves as defined by user .

$x = \text{real} + \text{imaginary}\, i$

Color 3D

To create a Color 3D graph, select an equation of the form z = f(x, y) and choose the fourth sub-option of the third option on the Graphs menu (or press Command-K):

$z = x \sin(y)$



The default description of this plot is shown in the graph details:

Surface at (x,y,z) where x = west ... east and y = south ... north ;
Opaque surface has mesh and is shaded using Custom coloring;
z = bottom ... top selects from the color scheme Oregon .

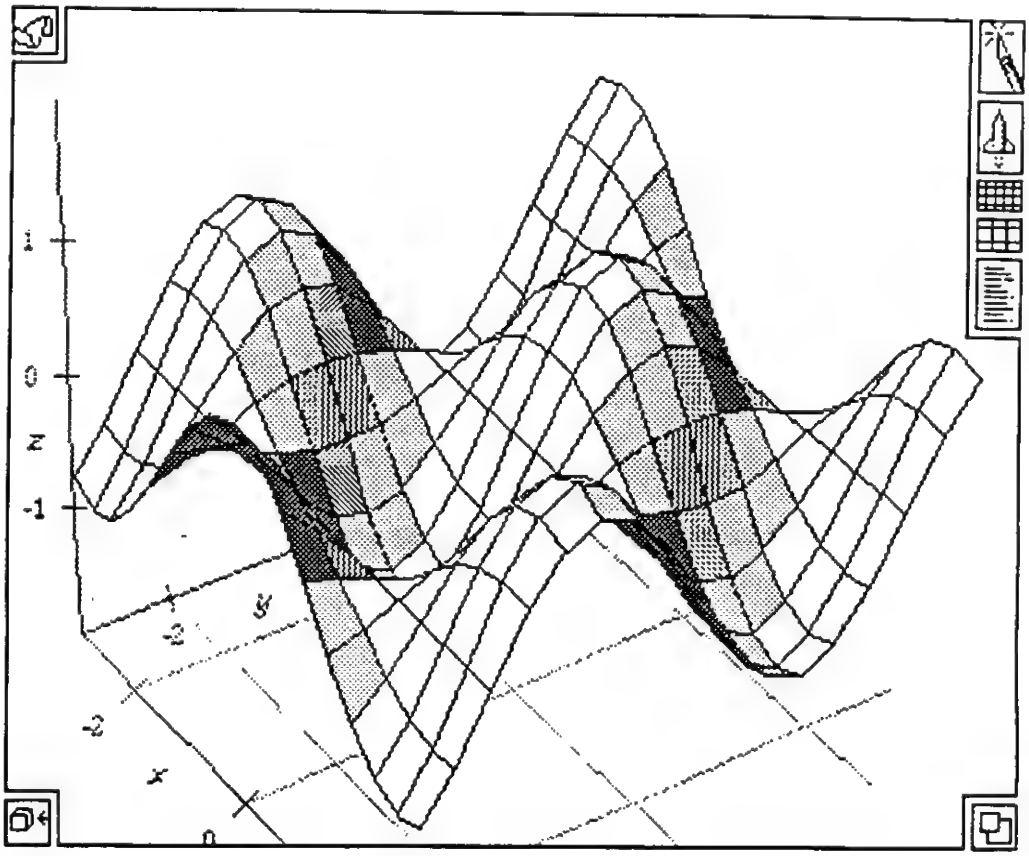
Color 3D graphs create Surface plots with Opaque optical characteristics, and use the Custom color interpretation.

Custom color provides seven different color schemes: Arizona, Kansas, Louisiana, New York, Oregon, Wyoming, and Virginia. Except for Kansas, all Custom color schemes range from black to white through a fixed set of three different colors. Kansas uses five colors and no black or white. Naturally, these all appear as shades of gray on a black and white monitor or on the printed page (unless you use a color PostScript printer).

For this graph, the resolution was increased from the default of eight to 16, so there are seventeen vertices from west to east and from south to north.

Illuminated 3D

To create an Illuminated 3D graph, select an equation of the form z = f(x, y) and choose the fifth sub-option of the third option on the Graphs menu (or press Command-U):



The default description of this plot is shown in the graph details:

Surface at (x,y,z) where x = west ... east and y = south ... north ;
Illuminated surface has mesh and is shaded using Solid coloring;
White is the solid color.

Illuminated 3D graphs generate a Surface plot with Illuminated optical characteristics and uses the Solid color scheme (a single color is used for all panels). White as the default color. The surface is displayed with a mesh. The illumination for an Illuminated Surface plot comes from over your left shoulder.

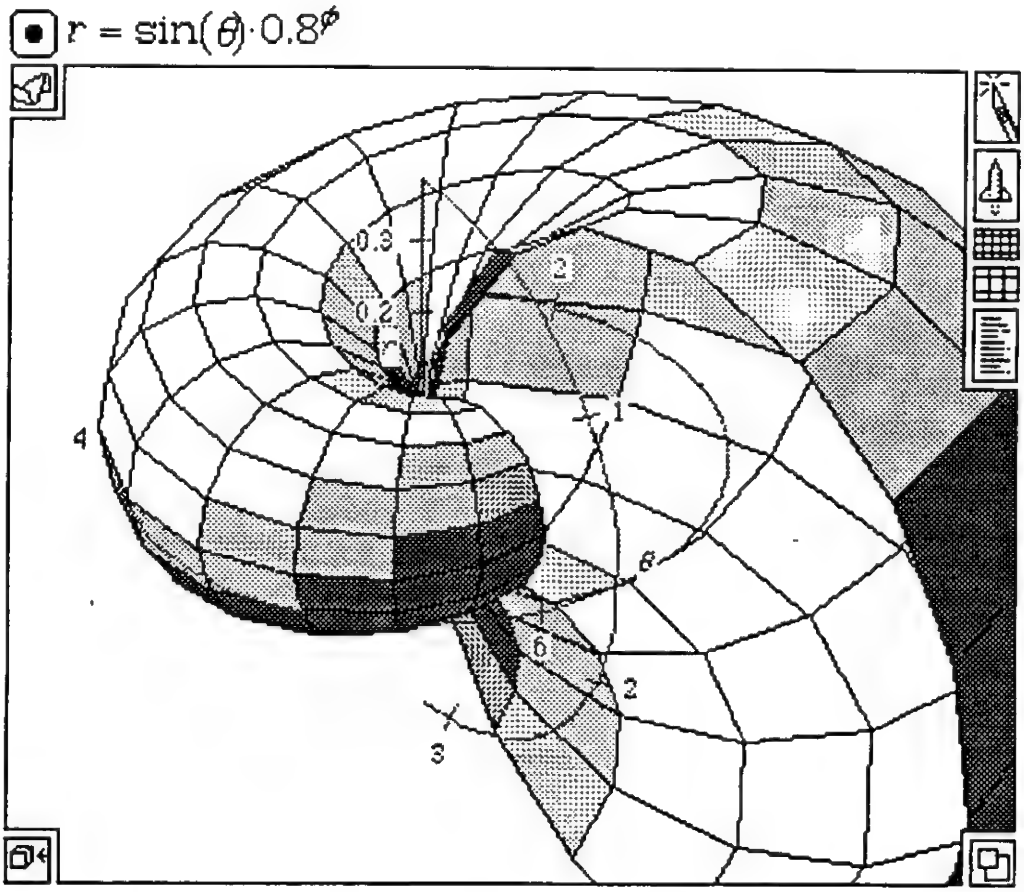
There are thirteen different colors that can be used with the Solid color scheme: Black, Blue, Green, Cyan, Red, Magenta, Yellow, White, Brown Orange, Light Gray, Gray, and Dark Gray. The fourteenth option on this pop-up menu is "Other...". This option brings up the Macintosh color wheel dialog box. From this dialog box, you can select a color from a rainbow of up to sixteen million colors, depending on your display device.

For more information about the Macintosh color wheel, consult the *Macintosh Reference* manual.

For this graph, the resolution was increased from the default of eight to 16.

Spherical 3D

To create a Spherical 3D graph, select an equation of the form z = f(x, y) and choose the sixth sub-option of the third option on the Graphs menu:



The default description of this plot is shown in the graph details:

Surface at FromSpherical(*r*, θ, φ) where θ = 0 ... π and φ = 0 ... 2π;
Illuminated surface has mesh and is shaded using Solid coloring;
White is the solid color.

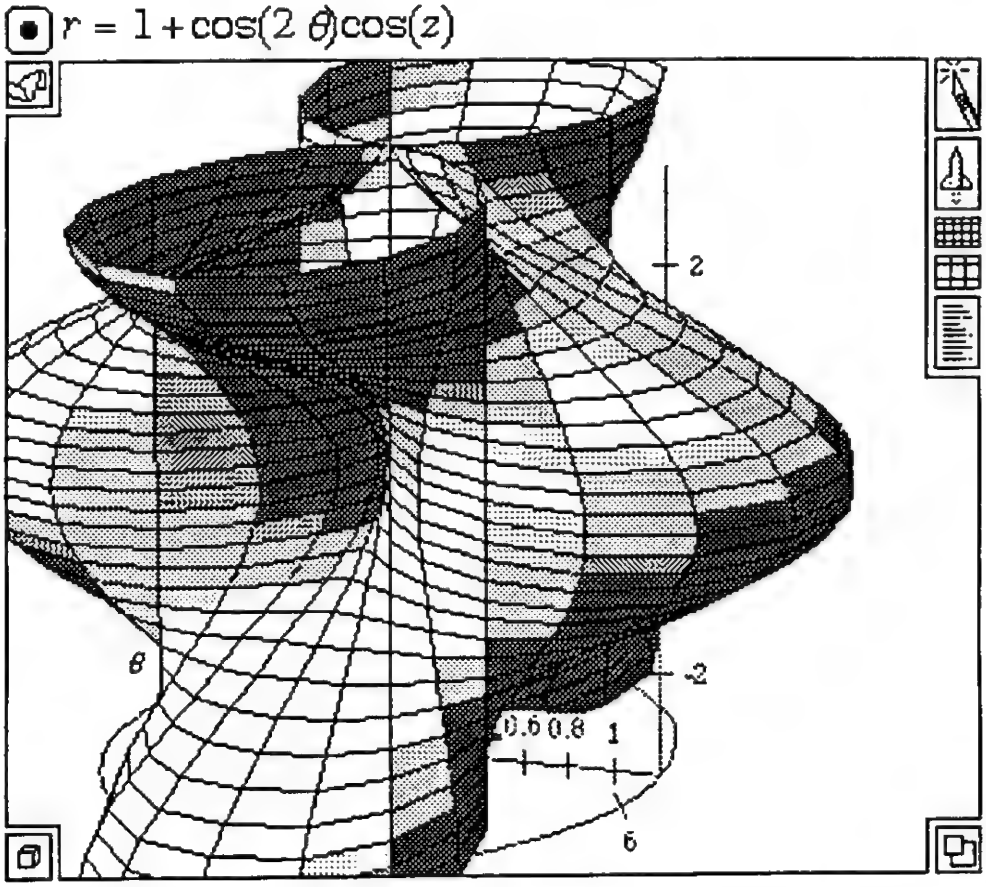
Spherical graphs generate a Surface plot with Illuminated optical characteristics similar to an Illuminated 3D graph except that the data points generated from the location vector are mapped through the function FromSpherical to create a spherical coordinate system. All characteristics can adjusted as in the Surface plot created for an Illuminated 3D graph.

For this graph, the domain for the third element in the location vector, φ, is from zero to 2π. The surface stops after completing one rotation "around the equator." To extend the surface deeper into the shell, increase this range (e.g., 0 ... 3π).

For this graph, the resolution was increased from the default of eight to 16.

Cylindrical

To create a Cylindrical graph, select an equation of the form z = f(x, y) and choose the seventh sub-option of the third option on the Graphs menu:



The default description of this plot is shown in the graph details:

Surface at FromCylindrical(r, θ, z) where θ = 0 ... 2 π and z = bottom ... top ; Illuminated surface has mesh and is shaded using Solid coloring; White is the solid color.

Cylindrical graphs generate a Surface plot with Illuminated optical characteristics similar to an Illuminated 3D graph except that the data points are mapped through the function FromCylindrical to create a cylindrical coordinate system. All characteristics can be adjusted as on an Illuminated 3D plot. For this graph, the resolution was increased from the default of eight to 32.

Surface Plot Propositions

Surface plot propositions describe the appearance Surface plots. These propositions contain several pop-up menus and values that can be easily adjusted to change many of the surface's characteristics. This section describes Surface plot:

- Optical characteristics
- Color schemes

Surface plots are created for Density, Complex 3D, Color 3D, Illuminated 3D, Spherical 3D, and Cylindrical 3D graphs. Each of these graphs generate a particular type of Surface plot. By default, these graphs generate the following plots:

Graph Type	Optical Characteristics	Mesh	Color Scheme	Color Setting
Color 3D	Opaque	Mesh	Custom	Oregon
Illuminated 3D	Illuminated	Mesh	Solid	White
Spherical 3D	Illuminated	Mesh	Solid	White
Cylindrical 3D	Illuminated	Mesh	Solid	White
Density	Opaque	No mesh	Gradient	Black to white
Complex 3D	Opaque	Mesh	Complex	N/A

The second line of a Surface plot proposition sets the fundamental graphic qualities of the plot. The first pop-up menu sets the optical characteristics and can be set to any of its four possible values:

Optical Characteristics	Appearance
Transparent	Surface appears like colored cellophane (white transparent in PostScript)
Translucent	Surface appears like colored wax paper (white transparent in PostScript)
Opaque	Surface appears made of solid matte colors with uniform diffuse illumination
Illuminated	Surface appears made of solid matte colors illuminated by a light source over your left shoulder

Transparent and Translucent are only supported on QuickDraw devices (monitors and printers). On PostScript devices (e.g., the LaserWriter), these plots are rendered as transparent and white. These two effects simulate clear and hazy stained glass; when printed on PostScript printers, all panels are clear with the white page (or what ever is behind) showing through.

Color Schemes

The last pop-up menu in the second line of a Surface plot proposition selects the color scheme, and creates a unique third line for each of its five settings: Solid, Gradient, Custom, Complex, and Direct. This section discuss each of these color schemes.

Solid

The third line of a plot proposition using the Solid color scheme contains a single pop-up menu that can be set to any color.

Surface at (x,y,z) where x = west ... east and y = south ... north ;
Illuminated surface has mesh and is shaded using Solid coloring;
White is the solid color.

In all cases, this menu defaults to white, but it can be set to thirteen named colors, and, using the last option, to any of 16 million colors:

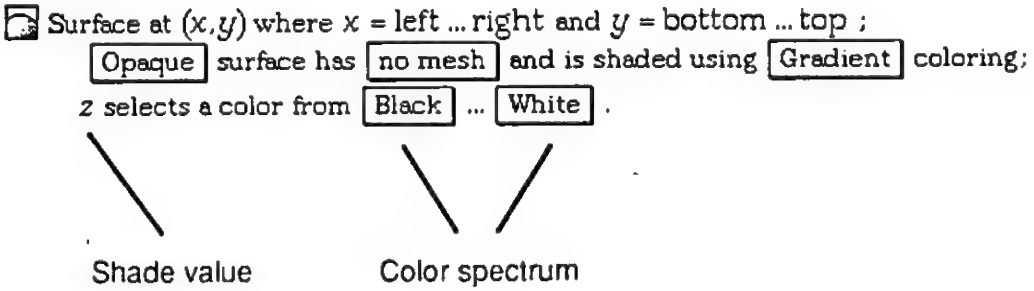
The last option, Other..., brings up the Macintosh color wheel dialog box. From this dialog box, you can select from up to 16 million colors, depending on your display device. For more information about the color wheel dialog box, consult the *Macintosh Reference* manual.

If the optical characteristics pop-up menu is set to Transparent, Translucent, or Opaque, all panels in the plot show the same selected color. Illuminated plots combine the lighting effects and the surface color; some panels appear darker than others, but the color value for all panels is the same.

Illuminated 3D, Spherical 3D, and Cylindrical 3D graphs generate plots with a Solid color scheme.

Gradient

The third line of a plot proposition using the Gradient color scheme contains a "shade value," and two pop-up menus that can be set to any color. The two menus are the same as the single pop-up menu found in Solid color Surface plots.



For the Gradient color scheme, a value is derived for each panel and that value specifies the color for that panel; the color of each panel is calculated independently.

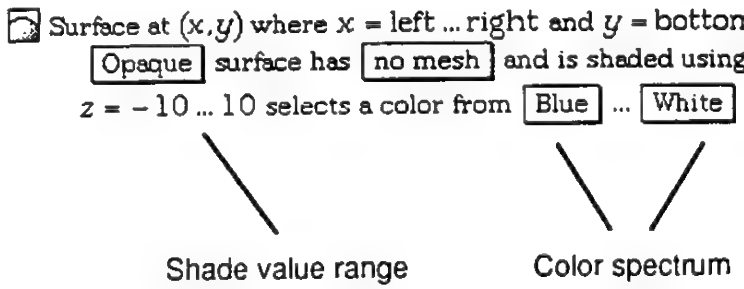
The shade value at the beginning of the last line (in this case, *z*) specifies a color selected from the continuum ranging between the two colors. The default shade value for a Density 2D graph (the only graph that creates a gradient plot by default) is the dependent variable.

The default range for the shade value is 0...1. Where the value of that variable is less than or equal to zero, the first of the two colors is shown. Where the value of that variable is greater than or equal to one, the second color is shown. For surface panels that have values between zero and one, some mix of the two selected colors is shown.

You can replace the shade value with any expression. But only expressions that evaluate to real numbers between zero and one produce a shade between the two colors (unless you specify another range of values as explained below). If the expression evaluates to a

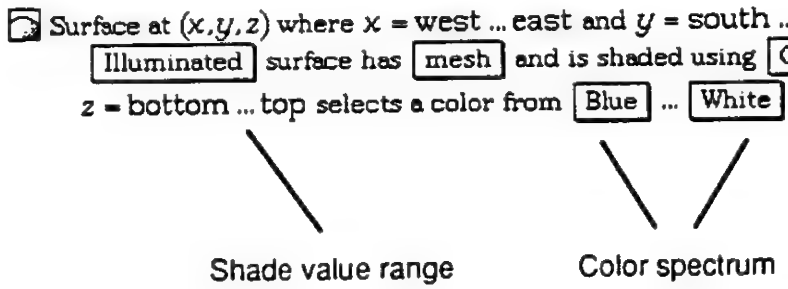
number outside the range 0...1, one of the two defining colors is used. If the expression evaluates to an unknown value (the question mark), or a complex number, the panel is shown as white transparent. If the expression evaluates to a matrix (or vector) an error is reported.

To specify a range of shade values other than 0...1, edit the proposition and insert a different shade value range:



In this case, −10 coresponds to the first color in the color spectrum and 10 coresponds to the second. All values within the given range are displayed in a shade between the two colors. In this example, any value greater than 10 is white and any value less than −10 is blue.

You can also set the range values to named variables or constants, such as bottom and top:



Values for the constants bottom and top are given in the graph bounds at the top of the graph details. With this set up, the panels at the top of a plot are white. Those at the bottom are blue. And those in between range from solid blue, through light blue, to white.

The shade value range can also be set to specific values. For example, suppose you set the shade value range to $z = 0 \ldots 3.14$. Any panels with z values less than zero are blue. All panels with z values greater than π (~3.14) are white. And all those in between range from blue to white.

The following table presents a set of possible shade values for the default range (0...1) and a description of the resulting colors. This table assumes that the first pop-up menu is set to red and the second to white.

Shade value	Resulting color
–2.00	Solid red
0.00	Solid red
0.30	Dark pink
0.80	Light pink
1.00	White
7.00	White
3+4*i*	Hollow (clear panel)
?	Hollow (clear panel)
(0.1, 0.2)	Error (a vector)

Density 2D graphs are the only graphs that generate plots with Gradient color schemes, but most plots can be easily adjusted to use this color scheme.

Custom

The third line of a plot proposition that uses the Custom color scheme contains a "shade value," and a single pop-up menus that can be set to one of seven state names: Arizona, Kansas, Louisiana, New York, Oregon, Wyoming, and Virginia. Each state name designates a particular group of colors. Kansas uses five colors, all other state names use three colors and black and white.

Surface at (x,y,z) where x = west ... east and y = south ... north ;
Opaque surface has mesh and is shaded using Custom coloring;
z = bottom ... top selects from the color scheme Oregon .

Shade value range — Color group

The Custom color scheme use a principal similar to Gradient color scheme. A value is derived from an expression for each panel and that

value specifies the color for that panel. The color of each panel is calculated independently.

If the shade value expression is between the specified minimum and maximum (in this case, bottom ... top), colors from the named group are used to color a particular panel. If the expression evaluates to a number outside that range, one of the extreme colors is used. (Except for the Kansas color group, the extreme colors are white and black.) If the expression evaluates to an unknown value (the question mark), or a complex number, the panel is shown as white transparent. If the expression evaluates to a matrix (or vector) an error is reported.

By default the range expression bottom ... top is associated with the shade value. But, as with the Gradient color scheme, this can be edited freely. The following table assumes that the range expression is set to 0 ... 1.

Value	Arizona	Kansas	Louisiana
≤ 0	Black	Violet	Black
.25	Brown	Blue	Blue
.50	Red	Green	Violet
.75	Yellow	Yellow	Red
≥ 1	White	Red	White

Value	New York	Oregon	Wyoming	Virginia
≤ 0	Black	Black	Black	Black
.25	Green	Blue	Violet	Brown
.50	Yellow	Green	Red	Gray
.75	Red	Yellow	Orange	Blue
≥ 1	White	White	White	White

As with the Gradient color scheme, the shade value range, by default, is given as bottom ... top. You can specify another range of values, to compress or expand the colors available for the panels of a particular Surface plot.

For example, to use a fine-grained set of colors from purple through wine red, set the color group to Louisiana, and change the shade value range to:

z = bottom – 9 ... top + 9.

The following table presents a set of possible shade values and a description of the resulting colors. This table assumes that the pop-up menu is set to the Oregon color group (the default), and that the value range is set to 0 ... 1.

Shade value	Resulting color
-2.00	Black
0.00	Black
0.10	Very dark blue
0.20	Dark blue
0.25	Bright blue
0.30	Blue with a bit of green
0.40	Sea green
0.50	Bright green
0.60	Chartreuse
0.70	Yellow with a bit of green
0.75	Bright yellow
0.80	Light yellow
0.90	Very light yellow
1.00	White
7.00	White
3+4i	Hollow (clear panel)
?	Hollow (clear panel)
(0.1, 0.2)	Error (a vector)

Color 3D graphs are the only graphs that generate plots with Custom color schemes, but most plots can be easily adjusted to use this color scheme.

Complex

Complex Surface plots use the magnitude and phase of the shade expression (in this case, *y*) to determine the brightness and hue of the surface panels. These colors are not adjustable. However, you can shift the magnitude or phase of the shade expression by multiplying it by a real or imaginary value (or both). If you replace the shade expression with explicit values (real or imaginary) all panels take on the same color .

Surface at (real, imaginary, $|y|$) where real = west ... east and imaginary = south ... north ;
Opaque surface has mesh and is shaded using Complex coloring;
y determines brightness and hue (from magnitude and phase).

The following table presents a set of possible shade values and a description of the resulting colors for the Complex color scheme.

Shade value	Resulting color
−100.00	Near white
−10.00	Light blue
−1.00	Blue
−0.30	Dark blue
−0.10	Navy blue
−0.01	Near black
0.00	Black
0.10	Dark brown
0.30	Mustard yellow
1.00	Yellow
10.00	Light yellow
100.00	Near white
−100.00*i*	Near white
−10.00*i*	Very very light green
−5.00*i*	Very light green
−2.00*i*	Light green
−1.00*i*	Sea green
−0.30*i*	Army green
−0.10*i*	Very dark green
0.10*i*	Dark violet
0.30*i*	Violet
1.00*i*	Light violet
10.00*i*	Very light violet
100.00*i*	Near white
−0.2 −0.2*i*	Steel blue
0.2 − 0.2*i*	Pea green
0.2 + 0.2*i*	Brick red
−0.2 + 0.2*i*	Purple
?	Hollow (clear panel)

Complex 3D graphs generate this type of plot.

Direct

You can only create Surface plots that use the Direct color scheme by modifying an existing Surface plot; none of the options on the Graphs menu produce this kind of plot.

Surface at (x,y,z) where x = west ... east and y = south ... north ;
[Illuminated] surface has [mesh] and is shaded using [Direct] coloring;
(z,x,y) = bottom ... top determines the color components [Red, Green, Blue] .

To create a Surface plot with a direct color scheme, choose Direct from the last pop-up menu in the second line of the plot proposition. Then change the single value at the start of the third line to a three vector (e.g., *z, x, y*). The elements of this vector determine the value of the three color components named in the final pop-up menu.

There are five possible interpretations of this vector:

Red, Green, Blue
Cyan, Magenta, Yellow
Hue, Lightness, Saturation
Hue, Saturation, Value
Y, I, Q

Red, Green, Blue
For this setting, each value in the three vector must be between 0 (black) and 1 (full color). Mixtures make all other colors. White is (1, 1, 1). This model is closest to the hardware; video screens usually have three color guns responsible for the red, green, and blue parts of the image. The RGB model is used for any hardware device that starts with black and adds light to it to get colors and ultimately white; so-called "additive" colors.

Cyan, Magenta, Yellow
Each value in the three vector must be between 0 (white) and 1 (full color). Black is (1, 1, 1) and white is (0, 0, 0). The CMY model is often used for printing processes that start with white paper and subtract light to make colors; so-called "subtractive" colors.

Hue, Lightness, Saturation
In this model, the hue is an angular quantity measured in radians. Zero is red, 2.1 is green, 4.2 is blue, and 6.3 (actually 2π) is red again. Lightness ranges from 0 (black) to 1 (white). The saturation also ranges from 0 (grays) to 1 (rich colors). If the lightness is in the middle of its range, then the saturation and hue become important. Extreme values for lightness reduce the effect of the saturation value.

Hue, Saturation, Value
The HSV model has largely replaced the HLS model. The Macintosh color wheel dialog box (used to set the color(s) for Solid or Gradient plots if you choose **Other**), uses this model. In this model, hue and saturation function the as in the HLS model. Value ranges from 0 (black) to 1 (rich colors and white).

Y,I,Q
This model is a matrix rotation in color space of the RGB model. It is used for broadcast television. The Y value specifies the brightness, the I value varies from cyan to red-orange, Q ranges from green to violet. I and Q assume values in the approximate range ±0.6.

The following tables displays example values the different models use for nine different colors. Values represented by an x can take on any value.

Color	RGB	CMY
Black	(0, 0, 0)	(1, 1, 1)
Gray	(.5, .5, .5)	(.5, .5, .5)
White	(1, 1, 1)	(0, 0, 0)
Red	(1, 0, 0)	(0, 1, 1)
Yellow	(1, 1, 0)	(0, 0, 1)
Green	(0, 1, 0)	(1, 0, 1)
Cyan	(0, 1, 1)	(1, 0, 0)
Blue	(0, 0, 1)	(1, 1, 0)
Magenta	(1, 0, 1)	(0, 1, 0)

Color	HLS	HSV	YIQ
Black	(x, 0, x)	(x, x, 0)	(0, 0, 0)
Gray	(x, .5, 0)	(x, 0, .5)	(.5, 0, 0)
White	(x, 1, x)	(x, 0, 1)	(1, 0, 0)
Red	(0, .5, 1)	(0, 1, 1)	(.3, .6, .21)
Yellow	(π/3, .5, 1)	(π/3, 1, 1)	(.89, .32, -.31)
Green	(2π/3, .5, 1)	(2π/3, 1, 1)	(.59, -.28, -.52)
Cyan	(π, .5, 1)	(π, 1, 1)	(.7, -.6, -.21)
Blue	(4π/3, .5, 1)	(4π/3, 1, 1)	(.11, -.32, .31)
Magenta	(5π/3, .5, 1)	(5π/3, 1, 1)	(.41, .28, .52)

For examples of graphs using the direct color scheme, see the "Direct Color Models" notebook on your distribution disks in the Graphics folder.

Animation

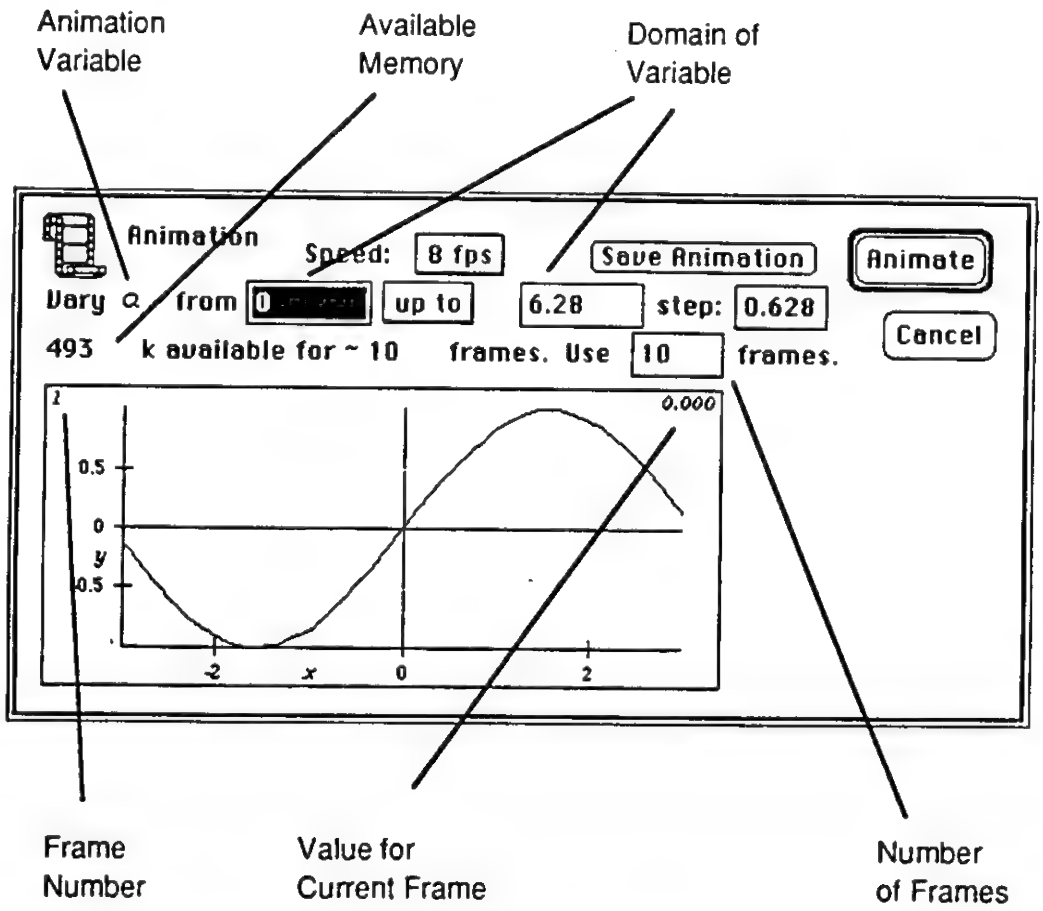
Theorist's animation facility lets you view a family of related graphs in quick succession. By creating several versions of a graph as one variable changes, saving these images, and displaying them in rapid succession, the animation creates the illusion of movement.

Animation is a technique for adding another dimension to a graph: time. When you animate a graph that displays the relationship between two variables, you introduce a third variable that changes through time. Similarly, a graph that displays the relationship between three variables, such as an Illuminated 3D graph, when animated, is actually displaying the relationship between four variables.

To create an animation, start with a two- or three- dimensional graph. The easiest way to animate a graph is to add a new variable, the animation variable (a, or any name of class Variable) to the original equation. This variable can be incorporated in the equation in any fashion. You can add it to any existing variable or expression, or multiply it onto any variable or expression, or use it alone (e.g., $\sin(a)$). The animation variable can also appear in any expression that affects the graph, including the graph bounds and all plot propositions. The variable can appear once, or several times in several expressions.

If you incorporate an undefined variable in an expression used to generate a graph, an error message appears stating that a value cannot be found for the new variable. This is perfectly normal. If you want to avoid this message, you can add an equation to temporarily define the the variable (e.g., $a = 1$), but this is not necessary.

Select the animation variable (or any expression containing the animation variable–but no other variable), and choose **Animate** from the **Graph** menu. The animation dialog box appears:



The exact size of the dialog depends upon the size of your graph. A copy of the viewport is displayed, but without the editing icons. You cannot edit a graph while animating it.

Across the top of the box are several pop-up menus and control settings.

The animation variable is displayed on the far left followed by the domain over which the animation variable will vary. This domain often defaults to run from zero up to 6.28 (2π). You can change both of these values, as well as the pop-up menu between them which indicates whether the number of frames should run "up to" (but not including) the maximum value or "through" (and including) that value.

Functions that are naturally periodic should use the "up to" setting. Images for the starting and ending values appear the same; you only want one frame for this value, not two. If your function is nonperiodic, use the "through" setting which pauses briefly between the last and first frame.

Below the animation variable is a number that indicates the amount of memory available for storing graph images. Available memory limits the actual number of frames that you can create. The number of frames to produce defaults to ten (regardless of the available memory). If more memory is available, you can increase the number of frames. If not enough memory is available, decrease the number of frames (otherwise frames are generated during the animation).

Increasing the number of frames decreases the step value and improves the visual resolution of the animation. The step value is determined by dividing the range of values by the number of frames. Changing the starting or ending values (or both), adjusts the step size but does not change the number of frames. Changing the step size directly does affect the number of frames.

Use the pop-up menu just above the range of variable values to set the speed of the animation play back. Speeds range from one frame every two seconds (1/2 fps), to 15 frames per second. One additional setting, "Fastest," displays the images as fast as your machine can move them from memory to the screen. For comparison, remember that most movies are displayed using 30 frames per second. These settings are approximate. If the images are very large, or you are running on a slower CPU, the time required to move the image from memory to screen may be greater than the designated increment between frames.

After selecting the number of frames to use, the range of the animation variable, and the playback speed, click on the "Animate" button (or press Return) to start the animation. The graph is rendered, displayed briefly, and saved in memory once for each frame. When all the images are rendered and saved, the animation starts in earnest. Frames flash on the screen at the requested speed, creating the illusion of motion.

To stop the animation, click the mouse anywhere or press any key (other than Return). Click on Cancel to quit the animation and throw away the rendered frames currently in memory. Changing the domain of variable values, or the number of frames used, also disposes of the stored images.

To save a series of frames, click on the "Save Animation" button. A dialog box appears asking for a file name. Saved animations use the PICS format and can be displayed with a wide variety of Macintosh animation programs that can read this file format. A simple program for viewing PICS files, called Projectionist, is included on the Theorist distribution disks. See Appendix B for instructions on using Projectionist.

Images saved in the PICS file are bitmapped images with the same "depth" (number of bits per pixel) as your Macintosh screen. (If you are using more than one monitor, the screen with the deepest pixel depth determines the depth of the saved images.) The amount of memory required for each frame is determined by the screen area it uses (the number of pixels), times the depth of your screen as set by the Macintosh Monitors Control Panel device.

No PostScript information is stored with an animation file. The type of PICT file generated (PICT 1 or PICT 2) is set by the Graph Preferences dialog box described later in this chapter.

On a Macintosh with a black and white monitor, a typical graph (two by three inches on screen) uses approximately 6.5K of memory. Whereas a Macintosh set up for 8-bit color (256 colors = 8 bits per pixel), uses approximately 52K for each such image. And a 24-bit color image of the same size uses more than 200K. Animating full color images is *very* memory intensive.

The complexity of a graph and the resolution (set with the editing icons) have no effect on the amount of memory needed for each frame, and little or no effect on the number of frames you can store. Complex graphs require longer processing, but a bit map of an intricate three-dimensional graph takes just as many bits as a graph of a single straight line (if the viewports contain the same number of pixels).

If you set the number of frames to a value greater than can be stored in memory, some frames are generated during play back, destroying the illusion of motion. However, you can save an animation with an arbitrarily large number of frames by pressing the "Save Animation" button. Each image is rendered in turn and saved to disk (rather than RAM). Use Projectionist to view an animation created in this manner.

If you don't have sufficient memory for the number of frames you want, you can attack the problem from three directions:

- Make the graph smaller (reducing the absolute number of pixels)
- Reduce the number of bits per pixel
- Allocate more memory for the Theorist application

Use the Monitors Control Panel device (CDEV) to change the number of bits per pixel. Changing from color to gray scale does not change the number of bits per pixel. The depth is the base 2 logarithm of the number of colors a monitor can display.

Monitors Setting	Depth (bits per pixel)
Black & White	1
4	2
16	4
256	8
Thousands	16
Millions	32

Printing and Exporting Graphs

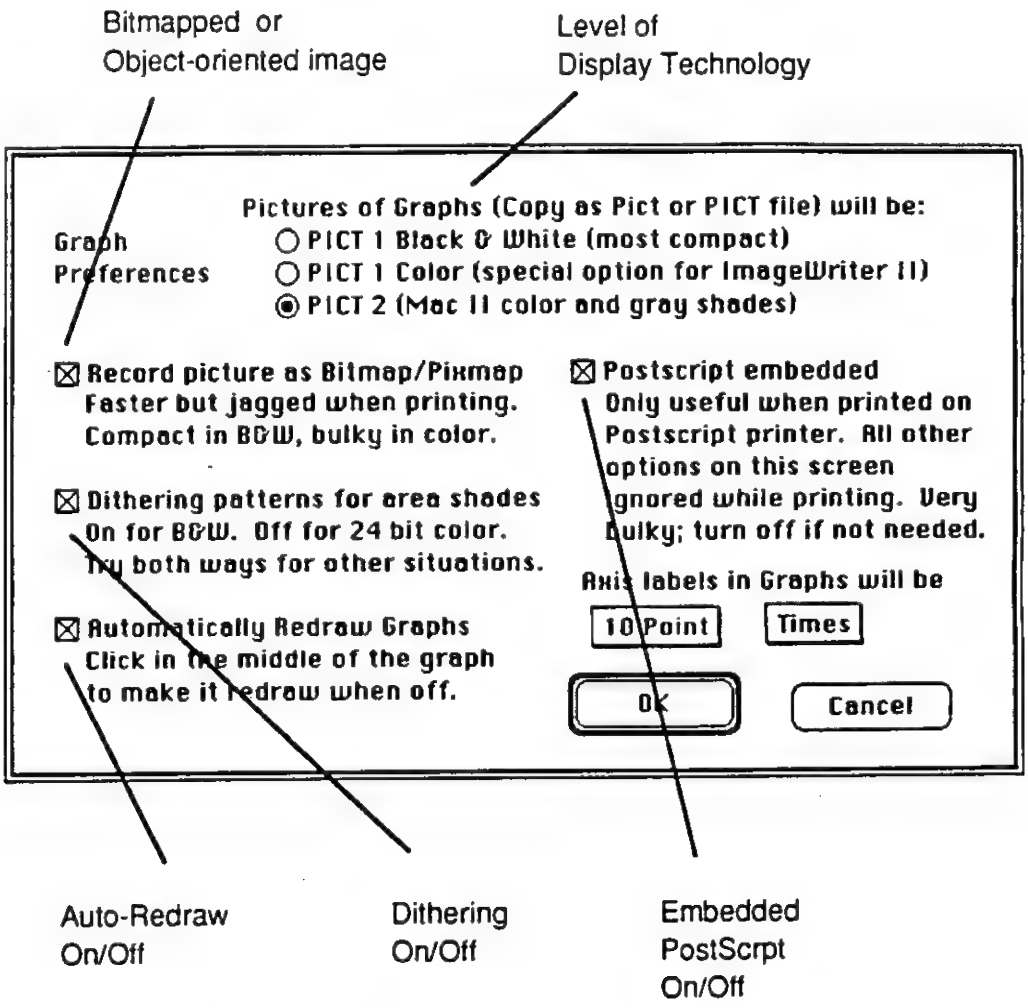
There are a variety of ways to produce the graphs you create in Theorist. Notebooks can be printed directly from the program on PostScript (e.g., LaserWriter) or QuickDraw (e.g., ImageWriter) printers. Graphs and graph data can also be copied to other programs in several formats.

This section discusses:

- Graph Preferences
- Printing graphs
- Exporting graph images
- Exporting graph data

Graph Preferences

The **Graph** option on the **Prefs** menu brings up a dialog box where you can specify various options that affect the appearance of graphs as they are displayed and printed, and the formats used to Copy graphs to other software packages.



Display Technology

At the top of the dialog box is a set of radio buttons for selecting the display technology used on your machine. PICT 1 displays graphs with one-bit pixels; either black or white. This setting is the only one available for non-Macintosh II machines. These images take up less memory (sometimes slightly less, sometimes enormously less).

Use PICT 1 Color only for printing color images on an ImageWriter II printer. If you use this option, turn Bitmap and PostScript off, and Dithering on. PICT 1 Color uses pre-Macintosh II QuickDraw routines, whereas the PICT 2 routines are specifically designed to display colors on Macintosh II generation machines.

On monitors that support color or gray-scale images, PICT 2 displays graphs with all the colors (or levels of gray) set by the Monitors Control Panel device.

Memory requirements for storing graph animations are *not* affected by this first setting. However, images and files produced with the Copy as PICT and Generate PICT commands are. In all cases, export images created with the PICT 1 setting are smaller than similar images created with the PICT 2 setting.

Avoid the combination of PICT 2 display technology, Bitmap off, and Dithering on. Many software applications do not support the codes that are generated by this combination, and may generate an image incorrectly.

Bitmapped vs Objects

The first check box in the left-hand column determines whether graphs are copied as bitmapped images or as an object-oriented description of the panels, vertex locations, and color settings. With black and white images, simple bit-by-bit storage takes the least memory. On the other hand, color images that are eight-bits deep (or more) are often cheaper to store as objects.

The amount of memory used for an object-oriented image depends on how many line segments (for Line plots) or how many individual panels (for Surface and Contour plots) are used to make up a graph. The size of an object-oriented graph depends on the resolution set for that graph.

The amount of memory used for a bit mapped image depends on the absolute size of the image (number of pixels) and the depth of each pixel (bits per pixel). The size of a bit mapped graph depends on the size of the graph.

If your images are only going to be displayed on screen, or exported to a low-resolution paint program, turn this option on. This saves the pixels of the image without the structure of the image, and is almost always faster and more compact.

If PICT 2 is set, this option generates a color pixmap image for display on a Macintosh II screen. All such images have the same pixel depth and color map as the monitor used to generate the images. If you are using multiple monitors of different types, graph images depend on the setting of the deepest screen.

Dithering

The second check box in the left-hand column is used to turn dithering on and off. Dithering is a technique of displaying a pattern of different colored bits (or black and white bits) to approximate an intermediate color (or shade of gray).

If PICT 1 is selected (one-bit black or white pixels), and dithering is off, each panel in a Surface plot is either all black or all white. If PICT 2 is selected, and dithering is off, each panel in a Surface plot is one solid color. If your display uses 24-bit color, turn this option off. The human eye can not distinguish between that many colors; dithering is not necessary.

You will probably want to keep dithering on for black and white monitors and off for 24-bit color monitors. For all situations in between try both ways to see which you like best.

You may also want to turn dithering off if you are printing on a non-standard printer, or are importing an image to another software package. Dithering patterns may be interpreted differently and this can create unintended effects.

Dithering usually increases the size of an image by a few to 40 percent.

Auto-Redraw

The third check box in the left-hand column does not affect graph printing or export. This option turns Auto-Redraw on and off. If you edit an equation used to create a graph, or the values in the graph details, the current display in the viewport is no longer valid. If Auto-Redraw is on, an image is regenerated and displayed immediately. If Auto-Redraw is off, the viewport is grayed out and displays the message: "Click to redraw." Make all your changes to equations and in the graph details, then click anywhere in the viewport.

Embedded PostScript

The last check box (on the right-hand side of the dialog box) determines whether PostScript descriptions are generated when printing a graph, or when you use Copy as PICT on the Edit menu, or Generate PICT file on the Graphs menu. QuickDraw is the native display technology for the Macintosh.

If you are printing on an ImageWriter or other QuickDraw printer, turn this option off to save processing time and memory. A PostScript description of an image is usually quite large, and unnecessary for these printers. However, this option must be on to print to a PostScript printer.

The PostScript description of a graph always contains full color information and overrides all other settings in this dialog box. The resolution of the printed image depends on the resolution of the printer. Postscript printers with resolutions greater than 1500 dots per inch (dpi), and full color PostScript printers are now available.

Note: Surface plots with Translucent or Transparent color schemes are displayed with hollow panels on PostScript devices.

Including a PostScript description doubles or triples the size of an image.

Axis Labels

These two pop-up menus determine the font and size of axis tick mark values and contour plot labels.

Printing Graphs

To print a graph as part of a notebook choose **Print** from the **File** menu. The Embedded PostScript option on the Graph Prefs dialog box must be checked if you are printing on a PostScript printer. Printing notebooks is described in the Introduction.

Exporting Graph Images

Graphs can be exported as graphic images in two different formats (PICT and EPS) and as lists or arrays of data point values.

Copy as PICT

Choose **Copy as PICT** on the **Edit** menu (or press Command-F) to copy a QuickDraw image of a selected graph (or proposition or expression) to the clipboard. You cannot use the standard Copy command to copy an image of a graph. These images can be pasted into any Macintosh program that accepts PICT format images.

The type of QuickDraw image created is determined by the settings in the Graph Preferences dialog box.

Generate PICT File

Choose **Generate PICT File** on the **Graphs** menu to create a named file that contains exactly the same information as Copy as PICT puts into the clipboard. The command brings up a dialog box that prompts you

for a file name. Generate PICT works better for larger images because copying images larger than 32k may not be possible, depending on the program the image is pasted into. However, not all programs can import a PICT file, whereas most all Macintosh programs can import a picture from the clipboard.

Generate EPS File

Choose **Generate EPS File** on the **Graphs** menu to create a named file that contains an Encapsulated Postscript description (and attached bit map) of the selected graph. The file is of type EPSF. The command brings up a dialog box that prompts you for a file name.

PostScript is a graphic page description language developed by Adobe Systems. Although originally developed for communications between computers and printers, it forms the basis of a standard file format for graphics called Encapsulated PostScript (EPS). An EPS file is a machine and resolution independent text file description of a graphic image. EPS files can be moved from one machine to another, or from one architecture to another, without loss of meaning.

You can import an EPSF file into certain Macintosh desktop publishing programs for high-quality results on any PostScript printer. The PostScript description contains full color information. These programs use the included bit map file to display a simplified image on screen.

Hold down the Option key while choosing the **Generate EPS File** option to generate a file of type TEXT, without any attached bit map. This file can be transferred to any other computer system that reads ascii text files.

In either case, the textual contents of the file follow Adobe Systems standards for Encapsulated PostScript. Coordinates are accurate to 40 microns. All surface colors are included, even on monochrome systems.

Warning: Do not use EPSF files with non-PostScript printers.

Exporting Graph Data

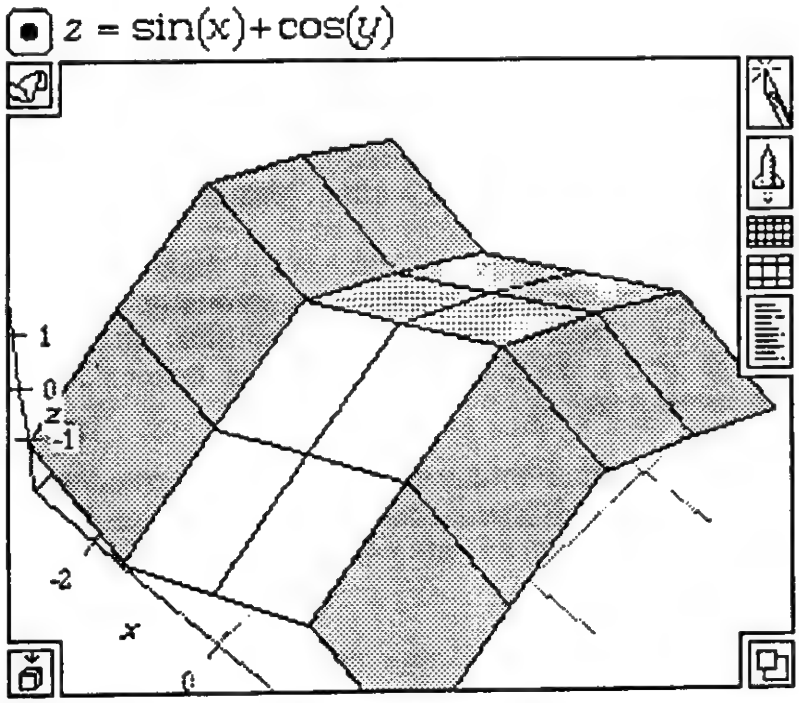
There are two commands for extracting data-point values from Line and Surface plots, Copy Heights Array, and Copy Points List. This section describes both of these commands.

Copy Heights Array

Use **Copy Heights Array** on the **Graphs** menu to generate a Tab- and Return-delimited list of the vertical height of each data point values in a Line or Surface plot.

For Line plots, this command produces a list of the vertical values (bottom ... top) for each data point. For a Surface plot, the command creates a table of values. To analyze these values in Theorist, you can Paste them into a matrix. You can also Paste the list into spreadsheet programs.

For example, consider the following simple Surface plot:



The resolution for this graph has been reduced from the default of eight to four to keep the example manageable. The plot consists of four panels on a side, or five vertices. The Copy Heights Array command produces a table of values, one value for the *z* coordinate of each vertex:

-0.8481	-0.05273	0.644	-0.05322	-0.8481
-1.491	-0.6948	0.001953	-0.6953	-1.491
-0.7422	0.05322	0.75	0.05322	-0.7427
0.005859	0.8013	1.498	0.8013	0.005859
-0.6367	0.1587	0.856	0.1587	-0.6367

Copy Points List

Use **Copy Points List** on the **Graphs** menu to generates a Tab- and Return-delimited list of the data point values, used to create a Line or Surface plot.

Each data point of a Line plot is represented by a pair of values for the *x* and *y* coordinates. Each data point of a Surface plot is represented by a trio of values for the *x*, *y*, and *z* coordinates.

Using the three-dimensional graph from the previous example, The Copy Points List command generates the following table, each of the twenty-five vertices represented by three values:

-3.375	-3.375	-0.8481
-3.375	-1.688	-0.05273
-3.375	0	0.644
-3.375	1.688	-0.05322
-3.375	3.375	-0.8481
-1.688	-3.375	-1.491
-1.688	-1.688	-0.6948
-1.688	0	0.001953
-1.688	1.688	-0.6953
-1.688	3.375	-1.491
0	-3.375	-0.7422
0	-1.688	0.05322
0	0	0.75
0	1.688	0.05322
0	3.375	-0.7427
1.688	-3.375	0.005859
1.688	-1.688	0.8013
1.688	0	1.498
1.688	1.688	0.8013
1.688	3.375	0.005859
3.375	-3.375	-0.6367
3.375	-1.688	0.1587
3.375	0	0.856
3.375	1.688	0.1587
3.375	3.375	-0.6367

Graph Programming

This chapter presents:

- An overview of graph programming
- A comparison of the details for Linear and Polar graphs
- Three examples of creating new graphs:
  - From scratch
  - By modifying variable dependencies
  - With spherical Contour plots

Overview

Using the commands on the Graphs menu, you can create a wide range of two- and three-dimensional graphs with several different types of Line, Contour, and Surface plots. You can also add all types of plots to existing graph theories with the commands on the Graphs menu.

After a graph is created, you can use the plot propositions in the graph details to create plots of your own design. The commands on the Graphs menu create and augment graphs using a number of defaults. If you edit plot propositions directly, a number of error messages are displayed, often indicating that certain variables do not currently have values.

To add a new plot to a graph, Copy a plot from from a neighboring graph, or use the Add... commands on the Graphs menu. If an equation is selected when you choose one of these options, the equation is used to generate the new plot. If no equation is selected, an empty proposition of the selected type is added to the graph details, or a duplicate of an existing plot, or a dialog box may come up asking for the independent and dependent variables.

You should never have to declare any of the names for the graph bounds (left, right, north, west, top, etc). If such a dialog box does comes up, you are probably trying to Copy a plot between a two-dimensional graph theory and a three-dimensional graph theory. This is rarely the best way to add a plot to a graph theory. Copying between graph theories of the same type works fine, as do the Add ... commands on the Graphs menu.

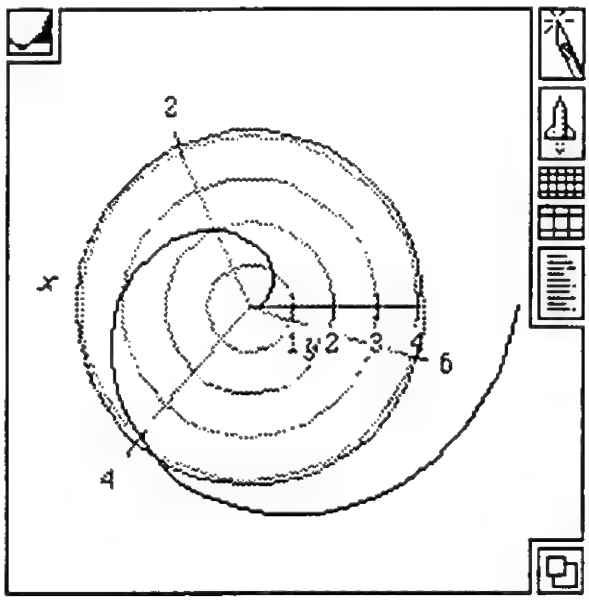
Most two- and three-dimensional graphs are created with all of the necessary name declarations in the Declarations comment near the top of the details box. However, if you are trying to make a polar, cylindrical or spherical plot in a rectangular graph theory, you need to declare the name "radius." Once declared, put the name declaration in the graph details, not in the Declarations comment for the main theory.

When you Cut, Copy, and Paste name declarations, you may get error messages declaring inconsistencies or unavailable values.

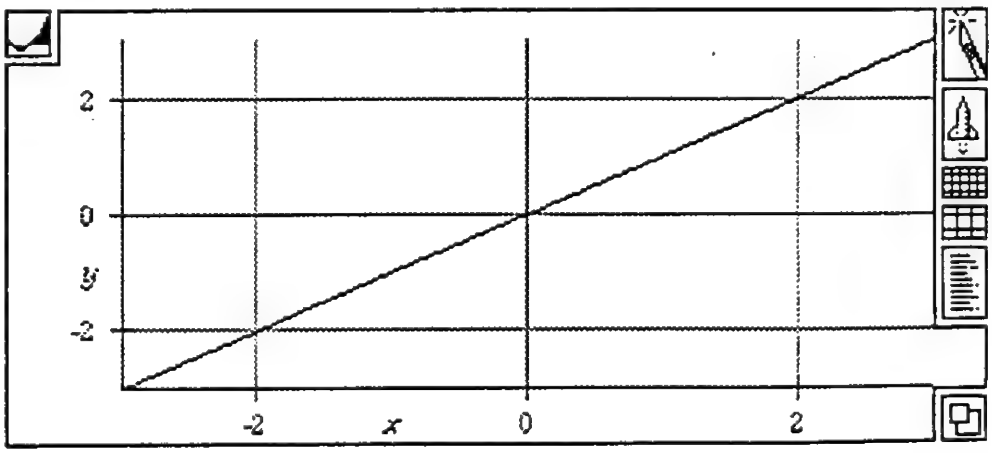
Comparing Graph Details

To gain a better understanding of the graph details, consider two graphs, one polar, one linear, created from the simple equation $y = x$.

Polar



Linear



The graph details for these two graphs appear as follows.

Polar

```
-3 ... 6.5 = left...right          True Proportions
-5 ... 2 = bottom...top      cropped Moderately

Declarations

Line at FromPolar(y,x) where x = 0 ... 2π with a normal line.
Axis at FromPolar(y,0) where y = 0 ... radius labeled y on this side.
Axis at FromPolar(radius,x) where x = 0 ... 2π labeled x on this side.
Grid lines at FromPolar(y,x) where y = 0 ... radius for each value
    x = 0 ... 2π separated by 0.
Grid lines at FromPolar(y,x) where x = 0 ... 2π for each value
    y = 0 ... radius separated by 0.
```

Linear

```
-3 ... 3 = left...right            Stretch to Fit
-3 ... 3 = bottom...top      cropped Moderately

Declarations

Line at (x,y) where x = left ... right with a normal line.
Axis at (x,bottom) where x = left ... right labeled x on this side.
Axis at (left,y) where y = bottom ... top labeled y on other side.
Grid lines at (x,y) where y = bottom ... top for each value
    x = left ... right separated by 0.
Grid lines at (x,y) where x = left ... right for each value
    y = bottom ... top separated by 0.
```

For the polar graph, y is the radius (r) and x is used as the angular value (often θ).

Location Vectors

In the polar graph, the location vectors for all plots (including the Axes and Grid lines) are mapped through the FromPolar function. The native coordinate system of a two-dimensional graph theory is rectangular; FromPolar is used to convert the coordinates. This function is available as a predefined function. If you create a polar graph by selecting the menu option, a dialog box may come up asking if you want to declare this function as a predefined function. The name declaration is included under the declarations comment at the top of the current notebook.

You can use working statements to create other mapping functions of your own design (e.g., FromHyperbolic) and use them in the same fashion.

Graph Bounds

Graph bounds are always in the native rectangular coordinates, even if the displayed graph uses another coordinate system. Whenever a graph is created, a set of default graph bounds is generated based on the function being graphed.

You can change the graph bounds with the Knife and the open-hand pointer, or by editing the values in the graph details. If you want to look at the upper right-hand quadrant of a polar graph, set the bounds to 0 ... 1 = left ... right and 0...1 = bottom ... top.

Ranges and Domains

In polar coordinate systems, it is customary to put the dependent variable (y or r) before the independent variable (x or θ). In rectangular coordinate systems, the order of the variables is reversed. The independent variable (x) comes first.

The domain of the independent variable (x) in a linear graph is, by default, dependent on the graph bounds. The Line plot in the above graph derives x values from the range left ... right, whereas the Line plot in the polar graph uses a "hard-wired" domain of 0 ... 2π. Scrolling either plot with the open-hand pointer changes the graph bounds, but the domain of a polar plot is not affected by this change.

In a polar plot, the domain of the independent variable "wraps around" on itself. By default, the domain is one revolution from zero to 2π, measured in radians. The Axes and Grid lines plots also use this domain. To change the domain in polar plots you have to edit these values directly; they are not dependent on the graph bounds as in linear plots. You must select the values and enter new numbers (or names or expressions that evaluate to numerical values).

The range of the dependent variable in a both graphs is dependent on the equation used to generate the graph. In a linear graph, these values are displayed along the vertical (bottom ... top) axis. In a polar graph the dependent variable is displayed along the radius.

Other differences

The name "radius" is a defined constant created and declared for all polar (and spherical and cylindrical) graphs. Its name declaration is placed in the Declarations comment at the top of the graph details. The value assigned to this constant is dependent on the logical size of the graphing area. If you zoom out the radius increases and a new set of Axes and Grid lines are displayed using the new value.

The default aspect ratio for polar graphs is True Proportions. For linear graphs, the default is Stretch to fit.

Example Graphs

This section describes how to create three different new graphs, including:

- Creating a graph from scratch
- Creating a new graph by reordering variable dependencies
- Creating a spherical Contour plot

Creating a Graph from Scratch

To create a non-normal graph, such as a three-dimensional Line plot, start by creating the type of graph theory you want as a shell. For example, to create a Line plot spiral in three space proceed as follows.

Enter an equation in three variables, such as z = x + y. Select it and create an illuminated 3D graph. Open the graph details and delete the Surface plot.

To insert a parametric Line plot into this empty shell, choose **Add Line Plot** from the **Graphs** menu. Accept whatever default dependency relations are displayed in the dialog box. Set the new Line plot to draw a normal line, and change the location vector to (*x*, *y*, *z*), and the domain to *t* = –10 ...10:

$\sim$ Line at (x,y,z) where $t = -10 \ldots 10$ with a [normal] line.

Then, outside the graph, create the following working statements:

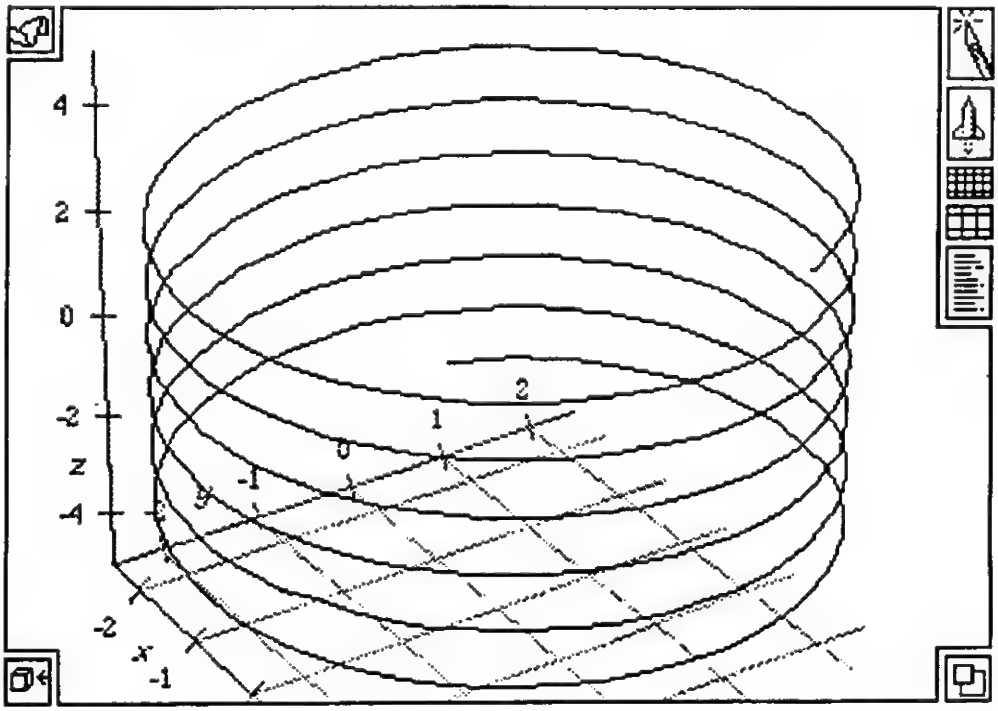
$x = 3\sin(2t)$

$y = 3\cos(2t)$

$z = \frac{t}{3}$

Depending on the equation used to create the three dimensional graph theory, you may need to adjust the graph bounds and the viewport controls. Set the aspect ratio to Stretch to Fit, and adjust the graph bounds to –2.5 ... 2.5 for the two horizontal coordinates (west ... east and south ... north), and –5 to 5 for the vertical coordinate (bottom ... top).

With a little rotation, the graph should look like this:



Using this technique, you can insert a Line, Contour, or Surface plot into any two- or three-dimensional graph theory.

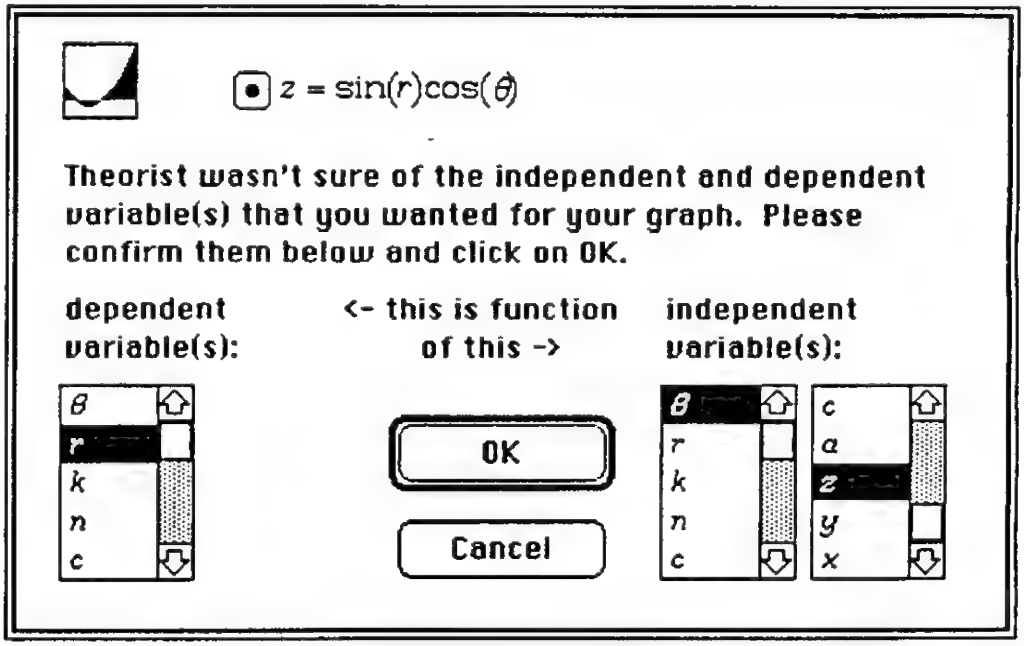
Reordering Dependencies

Rather than creating a plot completely from scratch, you can also reorder the variable dependencies of an existing plot to create a new plot. For example, to create a "disk plot" from a cylindrical plot, proceed as follows.

Enter the equation:

$z = \sin(r)\cos(\theta)$.

Select the equation and choose **Cylindrical 3D** from the **Graphs** menu. Declare all undeclared variables as "User Defined." When the dialog box for declaring the dependent and independent variables comes up, change the dependent variable to *r*, and the two independent variables to θ and *z*:



An error message declares that no value is available for *r*. Dismiss this message. By reordering the variables at this point, you create the desired spacial orientation. The domain of the surface plot can then be easily adjusted later.

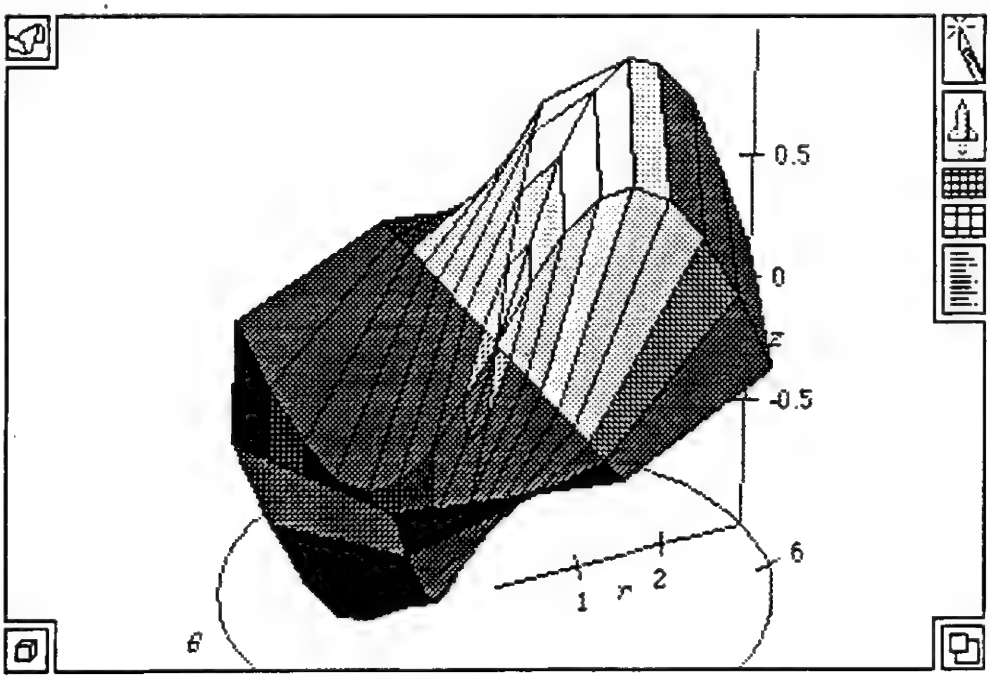
The graph displays the same error message. Open the details and change the domain of the Surface plot from z = bottom ... top to r = 0 ... radius. The last plot in the details (an axis plot) should use this as its domain; you can Copy it from there. The new plot proposition is:

Surface at FromCylindrical(r, θ, z) where $\theta = 0 \ldots 2\pi$ and $r = 0 \ldots$ radius ;
Illuminated surface has mesh and is shaded using Solid coloring;
White is the solid color.

Adjust the aspect ratio to Stretch to Fit, and change the bottom ... top graph bounds to a range of –1 to 1. And, for a final touch, change the color scheme pop-up menu to gradient, and use z = bottom ... top as a shade value expression:

Surface at FromCylindrical(r, θ, z) where $\theta = 0 \ldots 2\pi$ and $r = 0 \ldots$ radius ;
Illuminated surface has mesh and is shaded using Gradient coloring;
z = bottom ... top selects a color from Black ... White .

The graph, displaying a disk plot rather than a cylindrical plot, should look like this:



Spherical Contour Plots

Spherical contour plots are useful for plotting a quantity that varies over the surface of a sphere, such as the radiation pattern of an antenna or the temperature of the surface of a planet.

This type of graph takes particular advantage of a computer's interactive capabilities. On paper you can only see one side of the graph, whereas you can freely rotate the screen image. Unfortunately, due to the intense numerical calculations required for such an image, the display is quite slow on the slower Macintosh computers.

To create a simple spherical contour plot, proceed as follows.

Enter the equation you want to plot and a dummy equation for the radius (z):

$$r = \sin(2\theta)\sin(2\phi) + \theta$$
$$z = 9$$

Select the second equation and choose **Spherical 3D** from the **Graph** menu's **z = f(x,y)** submenu. At the dialog box for declaring the independent and dependent variables, choose z as dependent and θ and ϕ as the independent variables, in that order.

Delete the second equation ($z = 9$). Open the graph details and delete the last axis. Change the first element of the location vector of both remaining axis plots from radius to 11:

Axis at FromSpherical$\left(11, \frac{\pi}{2}, \phi\right)$ where $\phi = 0 \ldots 2\pi$ labeled ϕ on [other side].

Axis at FromSpherical$(11, \theta, 0)$ where $\theta = 0 \ldots \pi$ labeled θ on [other side].

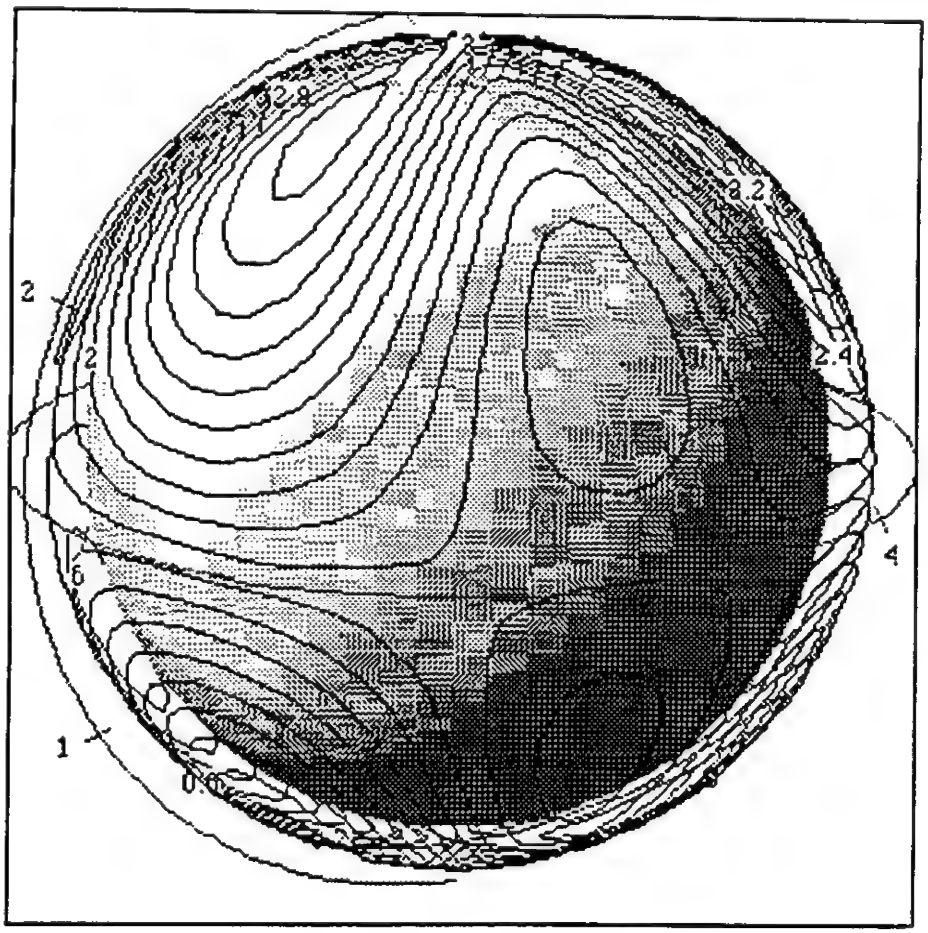
Edit the Surface plot proposition so that the location vector is $(9, \theta, \phi)$ rather than (z, θ, ϕ), and the mesh is off:

Surface at FromSpherical$(9, \theta, \phi)$ where $\theta = 0 \ldots \pi$ and $\phi = 0 \ldots 2\pi$;
[Illuminated] surface has [no mesh] and is shaded using [Solid] coloring;
[White] is the solid color.

Choose **Add Contour Plot** from the **Graphs** menu and accept the default dependencies. Copy the location vector and the two domain expressions from the Surface plot to the Contour plot (including the FromSpherical function). Increase the first element of the contour plot's location vector to 10, and change the "altitude" expression to r:

Contours at FromSpherical(10, θ, ϕ) where $\theta = 0 \ldots \pi$ and $\phi = 0 \ldots 2\pi$;
normally spaced contours of r.

Increasing the resolution from the default of 8 to 64, and turning Show Icons (on the Prefs menu) off, should create a graph that looks like this:



Editing

Editing mathematical expressions is a distinct art. Theorist incorporates a number of powerful conventions to simplify the modification of expressions.

This chapter provides a description of:

- The structure of mathematical expressions
- Using the palette to enter expressions
- An example sequence of deletions
- Techniques for selecting and editing:
  - Propositions
  - Expressions and sub-expressions
  - Multiple expressions
  - Names
  - Matrices and vectors
  - Comments
- Examples of creating and modifying:
  - Simple expressions
  - Functions
  - Linear operators
  - Calculus expressions
  - Iteration expressions
  - Matrices and vectors
  - Other expressions

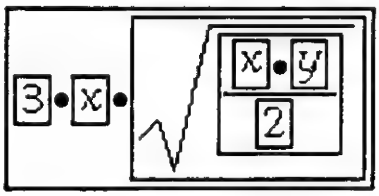
The Structure of Expressions

Creating and editing mathematical expressions is no ordinary task. When publishers typeset mathematics they often refer to it as penalty work; it takes twice as long as typesetting straight text. The computer scientist Donald Knuth went so far as to develop his own typesetting software (T_EX) to create his textbooks just the way he wanted.

The difficulty is inherent in the logical and visual structure of mathematical expressions. Unlike the linear boxcars of textual information, expressions are hierarchical structures. As an example, consider the following expression:

$$3x\sqrt{\frac{xy}{2}}$$

Most computer languages (Algol and its descendants, FORTRAN, Basic, Pascal, C, etc.) manage to reduce the two-dimensional complexity of such an expression to a linear string of characters. But in a graphic environment such as the Macintosh, it's possible to see and work with the expression more naturally, as a set of nested structures:



In Theorist, you can select and edit each nested object in an expression. Working with mathematical expressions is more complex than working with simple text. But that is hardly surprising. Mathematics is not used if an adequate verbal description is available.

This section discusses:

- Editing basics
- The Precedence Hierarchy and "Fortranish"
- Escape levels

Editing Basics

To edit an existing statement, the most important concept to remember is that each op *encloses* its operands. In fact, each statement, no matter how complex, is actually one or more expressions contained by a single op. This op is the highest-level structure in that expression. In an equation this op is the equals sign. To select an equation, click on the equals sign. You can use the highest-level op of any expression to select the entire expression.

To edit an existing expression, you'll get the most consistent results if you select the portion of the expression you want to edit, rather than trying to type into an insertion point.

To create a new statement, press Return. An empty statement appears, ready to be typed into:

If the selection point is in a proposition, the new statement appears immediately after that proposition. If there is no selection, the new statement appears at the bottom of the current notebook.

To create a comment proposition, press Shift-Return. The proposition appears at the bottom of the notebook or after a selected proposition. Text typed into a comment proposition wraps to new lines based on the size of the comment box. To create a hard carriage return (i.e., to force text to start on a new line) in a comment, press Shift-Return. Pressing Return alone creates a new statement (as it always does). If you intend to create a comment proposition but mistakenly create a new statement (by pressing Return), you can change the new proposition into a comment by pressing Shift-Return immediately.

Two commands on the Edit menu, Select Out (Command-A) and Select In (Command-E), are powerful tools for selecting expressions and subexpressions.

Select Out expands the current selection to the next largest structural level. If two or more selections collide, they undergo pairwise annihilation. That is, if an even number of selections collide, both selections disappear; if an odd number of selections collide, one survives and selects the larger expression

For example, in the expression $x + y + z$, if x and z are selected, and you invoke Select Out, both selections reach out to enclose the whole sum, but they collide and no selections remain. However, if x, y, and z are each individually selected, Select Out will select the entire sum, as if only one term was selected.

Select In selects all subexpressions of the currently selected expression. If you hold down Shift, Select In selects only the first subexpression.

Typing a closing parenthesis moves the cursor from anywhere inside an expression to the end of that expression.

If you select part (or parts) of an expression, the effect of the characters you type depends on whether the selections (and the entered characters) are ops or literal names (or numbers).

Most ops are created with punctuation symbols (e.g., +, *, /, \). Names are created with letters (Greek or Roman), and numbers are created with Arabic numerals (0, 1, 2, ...). Entering an op encloses any current selection(s) and *applies* the op to that selection (or selections). Typing in a name or literal number *replaces* the selection(s) with that name or number. Several examples are included in this chapter.

The Tab key moves the selection to the next question mark in the same proposition. If there are no question marks, the selection moves to the next subexpression of the same op. The arrow keys move the cursor in the designated direction. Due to the structure of propositions, this may not always be your intended direction.

If your keyboard does not have an Escape key (e.g., Mac Plus), use Enter instead.

Precedence Hierarchy and Fortranish

Theorist was designed to allow you to enter and edit equations as naturally as possible, interacting with them on screen as you would on paper. The mouse (and commands such as Select In and Select Out) let you work with expressions in a non-linear fashion. However, keyboard entry is quite linear and perhaps the fastest method for entering expressions.

Theorist offers two different methods of keyboard entry: one technique particularly designed for the graphic environment of the Macintosh, and another that follows the established conventions of computer programming languages (called "Fortranish").

In general, Theorist follows the traditional rules of operator hierarchy established by the Algol programming language (and used by its heirs: FORTRAN, C, BASIC, Pascal, etc.). The hierarchy that Theorist uses is shown in the following table. Each group of ops is at the same level.

Op Name	Character	Notation Form
Square Root	\ (back slash)	Prefix
Absolute Value	| (vertical bar)	Prefix
Subscript	_ (underscore)	Infix
Factorial	!	Postfix
Power (exponent)	^	Infix
Adjoint	Option-3	Postfix
Partial Derivative	Option-6	Prefix
Dot Product	Option-8	Infix
Cross Product	Option-7	Infix
Function Call	((after function name)	Infix
Division	/	Infix
Summation	@	Infix (multiple)
Multiplication	*	Infix
Negation/Subtraction	–	Prefix/Infix
Addition	+	Infix
Matrix Row	;	Infix
Matrix Column	,	Infix
Evaluate At	Option-|	Infix (multiple)
Range	Option-;	Infix
Integral	$ or ¢	Prefix/Infix
Pi Product	#	Infix (multiple)
Conditional	Option-Shift-|	Prefix
Relational Ops	= ≠ < ≤ > ≥	Infix

An op can either be entered before, in between, or after it's arguments. That is, ops are applied using prefix, infix, or postfix notation:

- Prefix ops are inserted before an argument. For example, to enter a square root, type a backslash and then the number.
- Infix ops are inserted between their arguments. For example, to enter a sum, type the first expression, a plus sign, and the second expression.
- Infix (multiple) ops have more than two arguments and are inserted between the first and second arguments. Use the Tab key to move to the other arguments. For example, the summation op is inserted between the increment variable and the first value for that variable. Press Tab to move to the next argument.
- Postfix ops are inserted after their arguments. For example, to enter five factorial, type the number five followed by an exclamation point.

Precedence rules are applied when an expression is potentially ambiguous. For example, the precedence rules dictate that the expression:

$5+7\times3$

be interpreted as five added to the product of seven and three.

If there is any conflict between ops of the same level, the ops bind from right to left. This is the reverse of the order often used. This means that $a/b/c$ indicates:

$$\frac{a}{b/c}$$

and not:

$$\frac{a/b}{c}$$

By default, Theorist breaks a few of the rules listed in the precedence table. Certain machine responses seem more natural in the graphic environment of the Macintosh. For example, if you create an exponent (using the circumflex, Shift-6), the cursor stays at the exponent level until you explicitly return it to the baseline (with the Escape key).

To create the expression x^{2+n}, using the default settings, type "x^2+n". Although this makes sense, it breaks with the standard expression hierarchy because the power op should bind more tightly than the addition op.

The following table shows how you would create the two expressions x^{2+n} and $x^2 + n$ with Fortranish on and off.

To Create:	With Fortranish off (the default)	With Fortranish on
x^{2+n}	x^2+n	x^(2+n)
$x^2 + n$	x^2 Esc +n	x^2+n

If you are familiar with entering equations in a programming language, you may want to turn Fortranish on. With this switch on, the hierarchy rules are followed explicitly. If these rules are not second nature for you, leave Fortranish off.

Expressions are copied to the clipboard in the Fortranish syntax. If you Copy an expression selection, the result is a text string representation of the expression that you can Paste into another program that accepts expressions (e.g., programming languages, spreadsheet programs). Use **Copy as PICT** (on the **Edit** menu) to create a bit-mapped copy of an expression.

Fortranish was designed to be as compatible as possible with the equation syntax of other computer systems. Most simple equations should move between systems with a minimum of editing.

If you are familiar with entering equations in Expressionist (the equation editor from Prescience), leave Fortranish off and enter equations as you would with that software. (Many of Expressionist's keyboard commands create the same structures in Theorist.)

All examples in this chapter assume that Fortranish is off, unless stated otherwise.

Escape Levels

Because Theorist does not always follow the precedence rules of equation entry, you need a technique for moving between the logical levels of an expression. Moving down from a superscript, or up from a subscript, or out of a square root, or out of a fraction's numerator or denominator, are all movements from one level of an expression to another escape level. The following ops form one distinct escape level:

- Power (exponents)
- Index (subscripts)
- Division (numerators and denominators)
- Square root
- Absolute value
- Integral
- Summation
- Pi Product
- Partial derivative
- Matrix (each element)
- Evaluate At

Each op that graphically encloses its subexpression makes an escape level.

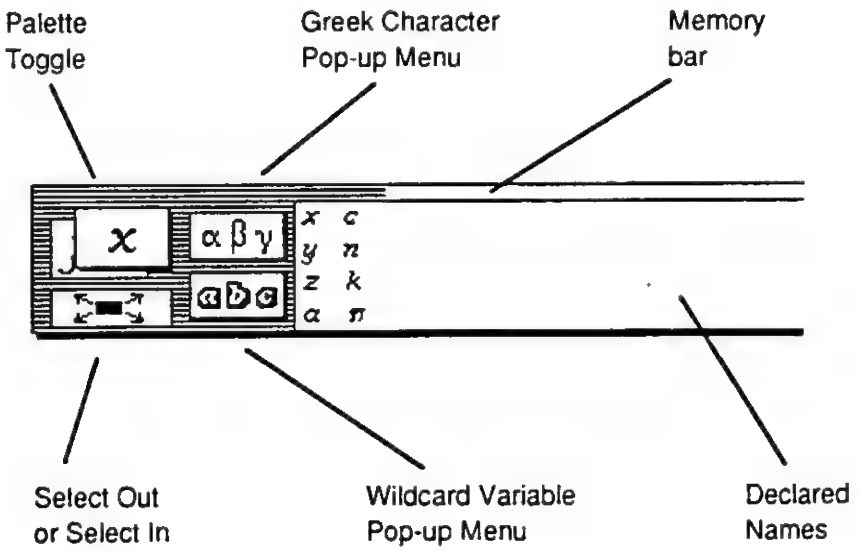
The easiest way to "escape" from a level is to press the Escape key. (Use the Enter key if your keyboard does not have an Escape key.)

Using the Palette

The "palette" is a rectangular window that appears above the current notebook. With the palette you can point-and-click to select the items you want to use to make up an expression without using the keyboard (except for numbers and new names). For some people, using the palette is easier than typing equations, and is probably faster when you first use Theorist. With practice, keyboard entry is usually faster.

The palette has two sides, shown below. The first is called the "variable palette," and the second is called the "function palette." You can switch between the two sides by clicking on the two overlapping icons on the left side.

The variable palette shows variables, constants, and operators that you can insert into an expression. Variables and the other items on this side of the palette always replace the current selection.

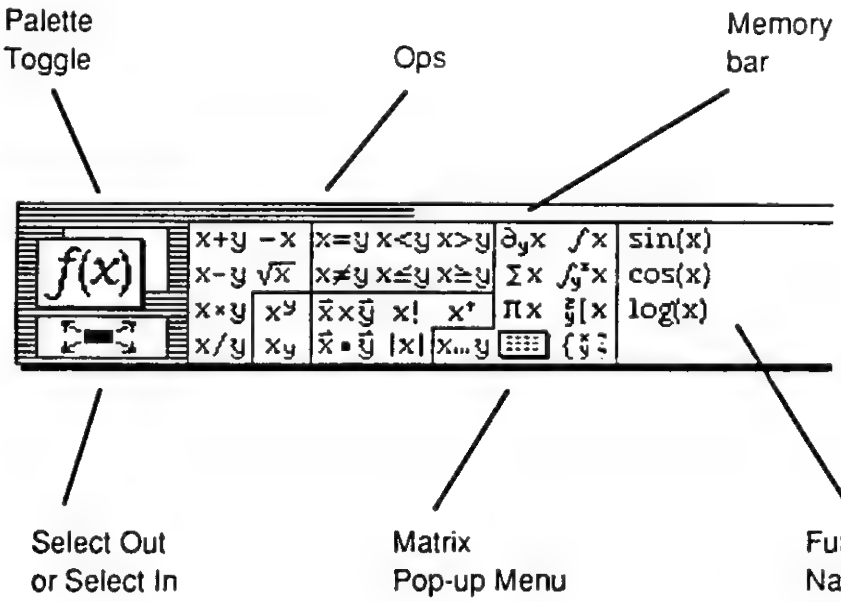


Use the αβγ pop-up menu on the left side of the variable palette to enter Greek characters. Choose multiple characters to create a name with more than one character. All other items on the variables palette insert just a single object.

Use the abc pop-up menu to insert wildcard variables.

The large area on the right of the variable palette displays the declared names of all variables, constants, and operators available in the current notebook.

The function palette displays ops and functions. All items on this side of the palette enclose an argument. If you click on an op or a function while something is selected, the selection is enclosed by that op or function.



Use the pop-up menu in the middle of the function palette to enter matrices of up to four-by-four elements. This pop-up menu also provides choices for inserting a new row or column in a selected matrix.

The area on the right of the function palette displays all declared functions. Click on a function name to enclose the current selection(s) with that function.

The memory bar indicates the amount of available memory. As memory is used, the bar extends to the right. As memory becomes available, the bar moves back to the left. If the bar moves very close to the right-hand edge of the palette, the menu bar flashes on and off indicating that the program is running very low on memory and may run out.

Click on the Select Out button to select a portion of an expression larger than the current selection. This button has the same effect as choosing **Select Out** from the **Edit** menu. If you hold down the Option key, this button, has the same effect as choosing **Select In**. Using Option-Shift and this button selects only the first subexpression of an expression.

To move the palette, click on any part of the palette that is not some kind of button and drag it to a new position.

To hide (or show) the palette, choose the **Show Palette** option on the **Prefs** menu. If this preference option is checked, the palette is displayed. To return the palette to its home position, hide it and show it again.

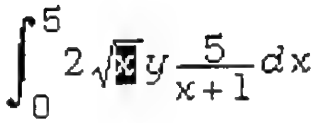
Example Sequence of Deletions

If you select an item in an expression and delete it, the item either disappears completely (if the selection is a string, a text string, or an N-ary selection), leaving an insertion point, or the item turns into a question mark. Deleting a question mark deletes its enclosing op, if any.

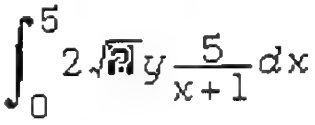
If an entire proposition is selected (including its icon), pressing delete removes the proposition and selects the proposition just above.

Pressing delete when there is no selection creates a new proposition at the bottom of the current notebook. Pressing delete again, removes that proposition and selects the proposition just above. If the next proposition (above) is collapsed, it is not selected. To delete a collapsed proposition, you must select it explicitly.

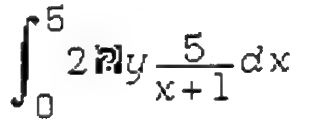
By repeatedly pressing Delete, you can unravel and delete an arbitrarily large expression from the inside out. In the following figure, *x* is selected in the initial expression, then Delete is pressed eight times.



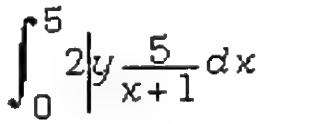
Selected x



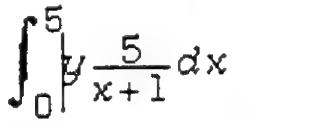
Deleted x



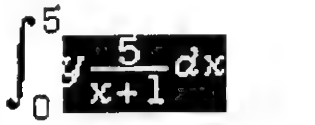
Deleted square root



Blinking insertion point



Deleted 2;
blinking insertion point



Integrand selected

Intagrand deleted

Integral selected

Integral deleted

Selecting and Editing Objects

This section discusses how to select and edit the following objects:

- Propositions
- Expressions and sub-expressions
- N-ary expressions
- Names
- Matrices and vectors
- Comments

All editing is based on selections. Whatever you type is applied to or inserted into the current selection. In addition, all manipulations work on the current selection. This section discusses the different types of selections and the kinds of editing you can do with them.

Selections work differently in different programs. In word processors, you select a sequence of characters, or possibly a sequence of paragraphs. In spreadsheets, you can select a set of cells which are a subset of a large matrix. In drawing programs, you select a set of objects. In outliners, you select hierarchical trees of topics.

In Theorist, you have a mixture of all of these types of selections. Not only that, but there can be multiple simultaneous selections. Each selection can be one of several types. Editing and manipulations are done on all selections simultaneously (to the limits of plausibility).

There are four ways to make a selection:

- Click
- Double-click
- Click-and-drag
- Double-click-and-drag

To create additional selections, hold down Shift as you click and drag or double-click. To unselect an expression in a multiple selection, shift click within that expression.

You cannot Copy from a multiple selection, but you can Paste into a multiple selection. Whatever you Paste is replicated in all selections.

Selecting and Editing Propositions

Click on a proposition's icon to select that proposition and all of its outlining subtopics (or daughters). Once selected, you can operate on all of them as a unit. These selections are useful for rearranging your propositions.

For example, if you press Delete, the selected proposition and all of its daughters are removed. If you Copy a proposition, all of its daughters are copied with it.

To duplicate a proposition, select it (by clicking on its icon), choose **Copy** from the **Edit** menu (or press Command-C), then select another proposition and choose **Paste**, also on the **Edit** menu (or press Command-V). The copied proposition appears after the selected proposition. Cut and Paste work the same way, except that the first proposition is deleted from its original position.

Selecting and Editing

To make an expression selection, click and drag within an expression. Only well-formed subexpression can be selected. For example, you can not select a square root without also selecting its argument.

There are many types of "insertion point" selections. These are analogous to insertion point selections in a word processor; they mark a place for insertion. For all insertion points, the selection is indicated by a blinking bar. All non-blinking selections enclose an expression.

Typing keystrokes and clicking on the palette inserts what you enter into the current selection(s). Op characters (mostly punctuation marks) always enclose the current selection (if it is not an insertion point). Name and number characters always replace the current selection.

Expressions can have the following types of selections:

- The entire expression
- A blinking cursor at the beginning
- A blinking cursor at the end
- Any type of selection in one or more subexpressions

You can use the mouse to click-and-drag to select any expression. A single click selects a single character name; a double click selects a multi-character name. You can also click on the distinctive part of an op to select the op and all its enclosed expressions:

- To select an equation, click on the equals sign.
- To select an integral, click on the integral sign.
- To select a fraction, click on the horizontal bar.
- To select an expression raised to a power, click just under the superscript.

There are some restraints on multiple selections. You cannot select something within an existing selection, nor can you select something that encloses an existing selection.

For example, if you select 456 in the expression 123 + 456 + 789, and then try to select the whole sum of three numbers by shift clicking and dragging, instead of making two selections, the two selections cancel each other out. The same thing happens if you try these two selections in the opposite order, selecting the sum and then attempting to select the number. Nor can you click-drag select two terms (e.g., 456 + 789) and then shift click to add the single term 123, or click-drag to add just the 2.

However, you can select each of the three numbers 123, 456 and 789 individually without conflict. Similarly, you can click-and-drag to select part of each term (e.g., 2, 5, and 89). But individual numbers (e.g., 123) cannot have two distinct selections.

Selecting N-ary Expressions

N-ary selections are selections that include one or more elements of a sum, product, or row vector. These expressions behave as if they were strings with each term acting as a single character. For example, in a sum of four terms, you can click and drag to select one, two, three, or all four elements:

$$1+2+3+4$$

You can also insert a blinking cursor between any of the elements (on top of one of the plus signs) or at the beginning or end of the sum. Use the arrow keys to insert the cursor at the beginning or end. If you select with a double-click, the selection is the individual term, not the sum. To select the entire sum, double click on one of the ops (the plus sign in this case), or double-click on one of the terms and start to drag across the terms.

Selecting Names and Numbers

Names and literal numbers in expressions behave as strings of characters as anywhere else in the Macintosh environment. For example, the name:

right

has six different possible insertion points: one at the beginning, one at the end, and four between the characters. You can also click-drag to select one or more characters. To select the name as a whole, double click anywhere within it.

Selecting Matrices and Vectors

Matrices are composed of two or more elements. Each of these elements can contain whatever selections are appropriate for that expression. Matrices as a whole have three special types of selections:

- Submatrix of the matrix
- Vertical insertion points between columns
- Horizontal insertion points between rows

To select an entire matrix, double-click on one element and drag, after the second click, to any neighboring element. To select a submatrix, click on one element, and drag to another element. To make a vertical insertion point, click between two columns. To make a horizontal insertion point, click between two rows.

To delete one or more rows from a matrix, select the entire row(s) you want from left to right and press Delete. Similarly, to delete one or more columns, select the entire column(s) from top to bottom and press Delete. If you select a submatrix, each element is replaced with a question mark. If you press Delete with a row (or column) insertion point selection, you delete the row (or column) to the left (or above) the insertion point, if any. If there is no row or column, Delete selects the whole matrix. Pressing Delete again removes the matrix.

To edit several matrix elements at once, click and drag over several elements to select them. Every keystroke, or palette entry, is applied to each of the selected elements. If you select the entire matrix (including its enclosing parentheses), whatever you enter replaces the matrix. To select all elements of a matrix individually, select the entire matrix then choose **Select In** from the **Edit** menu (or press Command-E).

If you Paste an expression into a submatrix selection it is replicated in each element, unless the clipboard contains a matrix. If you Paste a matrix (Cut or Copied from another matrix) into a submatrix selection, the matrix is pasted into the submatrix copying the elements one by one. If the copied matrix selection and the area you are pasting into have the same number of rows and columns, the elements are pasted in exact correspondence. If not, the pasted matrix repeats itself (if it is too short to fill up the selection) or terminates (if its too long).

For example, consider the following matrix:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \\ 21 & 22 & 23 & 24 & 25 \end{pmatrix}$$

If you Copy the element 5 and then select the submatrix from 8 through 25 and Paste, you produce this matrix:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 5 & 5 & 5 \\ 11 & 12 & 5 & 5 & 5 \\ 16 & 17 & 5 & 5 & 5 \\ 21 & 22 & 5 & 5 & 5 \end{pmatrix}$$

On the other hand, if you Copy the submatrix $\begin{pmatrix} 4 & 5 \\ 9 & 10 \end{pmatrix}$ and Paste it into the 8 through 25 submatrix, you produce the following:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 4 & 5 & 4 \\ 11 & 12 & 9 & 10 & 9 \\ 16 & 17 & 4 & 5 & 4 \\ 21 & 22 & 9 & 10 & 9 \end{pmatrix}$$

If you enter characters or Paste a selection into a vertical or horizontal insertion point, one or more new rows or columns are created.

Whatever you entered or pasted is inserted into the new elements as though they had previously existed and were selected before the Paste.

If an entire matrix is selected, entering characters or pasting a selection replaces the matrix.

Matrices work easily with spreadsheet programs. Copy a selection of cells out of a spreadsheet, then select a matrix or submatrix to Paste the cells into. If you select a whole matrix, the matrix adjusts to accept the cells from the spreadsheet as individual elements. If you select a submatrix, pasting works as with any other Copy and Paste operation.

Selecting Comments

A selection of text inside a comment is called a comment selection. You can have only a single selection in a comment. To edit comment propositions use Cut, Copy, and Paste, and the Delete key as you would in a word processor.

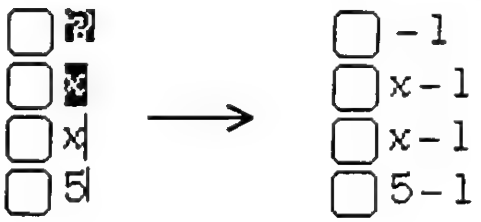
Examples of Expression Creation and Modification

Create expressions by typing or selecting items from the palette. The expression is entered wherever there is a selection. The interaction of the type of the selection(s) and what you enter determines the way the expressions appear. This interaction is rather complex, because of Theorist's power and flexibility. This section provides a number of examples of creating and editing:

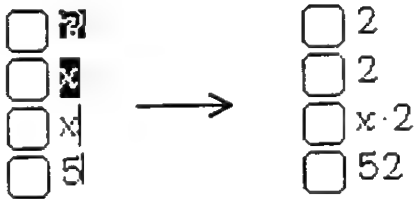
- Simple expressions
  - Numbers
  - Names
  - Ops
- Functions
- Linear operators
- Differential and Integral expressions
- Iteration expressions
- Matrices
- Other expressions

As an example of the interaction between types of selections and entered expressions, consider the following selection set:

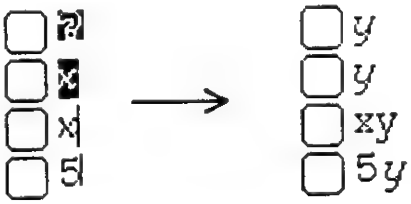
With the above selections, typing "-1" produces:



Typing "2" produces:



Typing "y" produces:



The new name xy, not *x* times *y*

Typed characters are interpreted as letters, numerals, or ops. Letters (Greek and Roman) form names of functions, variables, and constants. Numerals form numbers. Ops connect and enclose names and numbers to make expressions. Numerals can not be used as part of a name except as subscripts.

Examples in this section assume that all necessary predefined names have been declared and their associated predefined behaviors have been accepted. For more information about predefined names and behaviors, see the Expressions chapter.

Unless otherwise noted, the Fortranish option is off.

Simple Expressions

The following examples show how to create simple expressions, including:

- Numbers
- Names
- Ops

Numbers

Numbers are created by typing in numerals from the keyboard, with one exception. To create a number in scientific notation, enter the base value followed by the character "e". This produces a multiplication sign (×) followed by the number ten. Entered numbers become the exponent; entered letters are multiplied on as names.

To Create:	Type:	Comment
55	`55`	
−55	`-55`	The negation op encloses the number. Select a question mark before entering a negative value. Any other selection retains the selected expression and subtracts the number.
543.21	`543.21`	
.01	`0.01`	Enter the zero before the decimal to avoid confusion with the range op.
1.2×10^{6}	`1.2e6`	Use scientific notation.
1.2×10^{-4000}	`1.2e-4000`	Exponent can range between −4900 and +4900
1.23456789	`1.23456789`	Choose the display precision on Prefs menu. Nineteen digits are stored internally.
∞	Option-5	Infinity (a predefined name)
$3+4i$	`3+4i`	Complex numbers use the predefined name *i*
?	`?`	The question mark represents an empty or undefined value; select it and enter an expression.

Names

Names consist of one or more characters (Greek or Roman) typed in sequence. All capital and lower case letters can be used. Use the apostrophe (') for the prime mark. Use two prime marks in a row for double prime(''). Press Option-5 for the infinity symbol (∞).

Numerals can *not* be used as part of a name, except as subscripts.

Enter Greek characters by holding down Option and pressing the appropriate key, or select the character from the αβγ pop-up menu on the variables palette.

To Create:	Type:	Comment
x	`x`	Single character names are displayed in italics
sin	`sin`	Multi-character names are displayed in roman font
Re	`Re`	Names are case-sensitive; Re ≠ re
x'	`x'`	Use an apostrophe to create prime marks
ψ	Opt-`y`	Use the Option key to enter Greek characters
Ψ	Opt-Shift-`Y`	Use Option and Shift to enter upper-case Greek characters
ℵ	Opt-Shift-`A`	Use Option and Shift to create other symbols (see Appendix A)
g_h	`g_h`	Use the underscore character to create subscripts

(Continued)

To Create:	Type:	Comment
g_{sun}	`g_sun`	Use any name to create a subscript
g_2	`g_2`	Use any integer (±32000) for subscripts
$\aleph_0$	Opt-Shift-`A_0`	
$\mathbb{k}$	`?k`	Enter a question mark and any lower-case character (a - z) to create wildcard variables.

Ops

Ops are usually entered as punctuation marks. Any selected expression is enclosed by the entered op.

To Create:	Type:	Comment
$x + 10$	`x+10`	Spaces entered in sums are ignored
$5x$	`5*x`	
$5x$	`5x`	Concatenation acts as multiplication between numbers and names
$5(3x + 1)$	`5*(3*x+1)`	Enter multiplication symbols (*) explicitly, if you want to
$5(3x + 1)$	`5(3x+1)`	In most cases, multiplication symbols (*) are not required

(Continued)

To Create:	Type:	Comment
$5(3x+1)$	`5[3x+1]`	Use parentheses or brackets to enclose an expression
$5(3x+1)$	`5{3x+1}`	Or braces
$5(3x+1)$	`5(3x+1]`	Pairs of parenthetic marks do not have to match
xy	`x*y`	
xy	`x` space `y`	Indicate multiplication between names with an asterisk (*) or a space; concatenating names creates a new name (xy)
$x+\frac{1}{x}-2$	`x+1/x` Esc `-2`	
$x+\frac{1}{x}-2$	`x+1/x-2`	Fortranish on
$\frac{x+1}{x-2}$	`(x+1)/x-2`	
$\frac{x+1}{x-2}$	`(x+1)/(x-2)`	Fortranish on
x^2	`x^2`	Basic syntax
x^2	`x**2`	FORTRAN language syntax

(Continued)

To Create:	Type:	Comment
x^2+1	`x^2 Esc +1`	
x^2+1	`x^2+1`	Fortranish on
$(1+x)^2$	`(1+x)^2`	
x^{n+1}	`x^n+1`	
x^{n+1}	`x^(n+1)`	Fortranish on
x^{2n}	`x^2*n`	
x^{2n}	`x^(2*n)`	Fortranish on
$\sqrt{2}$	`sqrt(2)`	FORTRAN language syntax
$\sqrt{2}$	`\2`	Use the back slash character to enter square roots quickly
$\sqrt{x^2-1}$	`\(x^2 -1)`	Fortranish on. Use parentheses to enclose the argument. If not, the root encloses only the x^2
$\sqrt{\frac{x}{b_0}}$	`sqrt(x/b_0)`	FORTRAN language syntax

(Continued)

To Create:	Type:	Comment
$\sqrt{\frac{x+1}{x-1}}$	`sqrt((x+1)/(x-1))`	Use parentheses to specify the entire argument and the numerator and denominator
$n!$	`n!`	
$(2n)!$	`(2n)!`	Use parentheses to indicate a factorial expression with more than one element
$\frac{1}{n!}$	`1/n!`	Parentheses are not needed in this fraction; the factorial op binds tightly to the *n*
$\|x\|$	`|x`	Use a single vertical bar (|) to create an absolute value; don't type a closing vertical bar
$\|g_2 - g_1\|$	`|(g_2 Esc -g_1`	
$\|g_2 - g_1\|$	`abs(g_2 Esc -g_1`	FORTRAN language syntax
$x \bullet y$	`x Opt-8 y`	Use Option-8 to create a dot product
$x \times y$	`x Opt-7 y`	Use Option-7 to create a cross product
$A^\dagger$	`A Opt-3`	Use Option-3 to create an adjoint

The following examples show how to edit simple expressions.

To Change This:	To This:	Do This:
$5x$	$5xy$	Select *x*, type ***y**
$5x$	$5\frac{x}{y}$	Select *x*, type **/y**
$5x$	$\frac{5x}{y}$	Select $5x$, type **/y**
$5x$	$5(x+y)$	Select *x*, type **+y**
$5x$	$5\sqrt{x}$	Select *x*, type \ or Cmd-R
$5x$	$\frac{5}{x}$	Select *x*, Cut, Delete, type /, and Paste the *x* back in as the denominator

The minus sign does double duty as an unary minus op (–5) and as a sign indicating subtraction. If you type a minus sign (–), it is usually interpreted as a binary op. To create an unary minus, use Option-minus. If the selection is already negated, the ops cancel and the selection becomes positive.

To Change This:	To This:	Do This:
x	$x-y$	Select x, type –y
$x+y$	$x-y$	Select y, type Option-–
$x-y$	$x+y$	Select y or $-y$, type Option-–
$x+y$	$x+y+z$	Select y or $x+y$, type +z
$x-y$	$x-y+z$	Select $-y$, type +z
$x-y$	$x-(y+z)$	Select y, type +z
$5x$	$-5x$	Select $5x$, Option-–
$5x$	$-5x$	Click beginning of 5, type –
$5x$	$(-5)x$	Select 5, Option-–
$5x$	$5(-x)$	Select x, Option-–

Functions

The following examples show how to create various functions.

To create:	Type:	Comment
$\sin(x)$	`sin(x)`	The first parenthesis creates the function call op; the second escapes the function call op.
$\sin\left(\sqrt{\frac{x}{b_0}}\right)$	`sin(sqrt(x/b_0))`	The two closing parentheses escape the square root op and the sin function, respectively.
$\frac{\sin[x]}{\cos[x]}$	`sin(?x)/cos(?x)`	
$\sin^2 x$	`(sin(x))^2`	The form "$\sin^2 x$" is not recommended. This form changes to $(\sin(x))^2$ the next time you Simplify.
$\cosh(2x)$	`cosh(2x)`	
$\log(x)$	`log(x)`	log base 10 of x
$\ln(x)$	`ln(x)`	natural log
$\log_2(x)$	`log_2` Esc `(x)`	log base 2 of x
$\log_x(y)$	`log_x` Esc `(y)`	log base x of y
e^{ikx}	`e^i*k*x`	

Linear Operators

To multiply non-commutative operators onto each other, use the standard technique for multiplication: type an asterisk (*) or a space. Matrices are M-Linear operators and derivatives are D-Linear operators, as are any operators you create and designate to be M- or D-Linear operators. D-Linear operators may not be commutative or associative: if *A*, *B*, and *C* are D-Linear operators, the expression *A*B*C* is interpreted as *A*(*BC*), which is *not* the same as (*AB*)*C*.

The following examples show how to create various Linear operators.

To Create:	Type:	Comment
$A\,x\text{-}\lambda\,x$	`A x - Opt-1 x`	
$A\,x\text{-}\lambda\,x$	`A*x - Opt-1*x`	
$A^{-1}BA$	`A^-1 Esc B*A`	
$A^{\dagger}BA$	`A Opt-3 B*A`	

latrices

A matrix with a single row is displayed as a comma-separated list (a, b, c). Matrices with more than one row are displayed without commas.

The following examples show how to create various matrices and vectors.

To Create:	Type:	Comment
(1, 2, 3)	`(1,2,3)`	Use commas to separate elements
$\begin{pmatrix} a \\ b \\ c \end{pmatrix}$	`(a;b;c)`	Use semicolons to separate rows
$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$	`(1,2,3;4,5,6;` `7,8,9)`	Use commas to separate elements and semicolons to separate rows
$\begin{pmatrix} 0 & i \\ -i & 0 \end{pmatrix}$	`(0,i;-i,0)`	
$\left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right)\begin{pmatrix} x \\ y \\ z \end{pmatrix}$	(∂x Esc , ∂y Esc , ∂z) `(x;y;z)`	Type Opt-6 to create a partial derivative
A_n	`A_n`	Single subscripts indicate a row of a matrix or an element of a vector
$A_{(n,m)}$	`A_(n,m)`	Two subscripts indicate a single element of a matrix
$(a, b, c)_2$	`(a,b,c)_2`	This expression simplifies to b

Differential & Integral Expressions

The following examples show how to create various calculus expressions. To create the partial differential op (∂) press Option-6. To create an integral without limits (∫), type a dollar sign, "$".

To Create:	Type:	Comment
$\frac{\partial}{\partial x}y$	∂x Esc *y	Enter an asterisk (*) to explicitly indicate multiplication
$\frac{\partial}{\partial x}y$	∂x Esc y	A space can also indicate multiplication
$\frac{\partial}{\partial x}x^2$	∂x Esc x^2	
$\frac{\partial}{\partial x}(x^2+2x)$	∂x (x^2+2*x)	
dx	d*x	Multiplication with an asterisk (*)
$d(xy)$	d(x*y)	
$\int xdx$	$x d x	Enter an integral without limits before its argument
$\int_0^{\infty} xdx$	x d x¢0 Tab Opt-5	Enter an integral with limits after its argument
$5\int_0^{\infty} xdx$	5(x d x¢0 Opt-5)	Use parentheses to indicate integral's argument
$\int_0^5 \sqrt{\frac{x-1}{x+1}}dx$	sqrt((x-1)/(x+1)) d x¢0 Tab 5	

The following examples show how to create derivatives from various expressions.

To Change This:	To This:	Do This:
y	$\frac{\partial}{\partial x} y$	Click at left of *y*; type Opt-6 **x**
$\frac{\sqrt{y}}{z}$	$\frac{\partial}{\partial x}\frac{\sqrt{y}}{z}$	Click at left edge of the fraction; type Opt-6 **x**
$\frac{\sqrt{y}}{z}$	$\frac{\sqrt{\frac{\partial}{\partial x} y}}{z}$	Click at left edge of the *y*, type Opt-6 **x**
$\frac{\partial}{\partial x}$	$\frac{\partial}{\partial x} y$	Select the partial derivative; type * y
anything	$\frac{\partial}{\partial x}$ anything	Enclose anything in parentheses; click on the left parenthesis; type Opt-6 **x**

Iterations

The following examples show how to create various iteration expressions.

To Create:	Type:	Comment
$\sum_{n=0}^{5} a_n$	`n@0` Tab 5 Esc `a_n`	Summations are multiplied onto their arguments (because they are D-Linear operators)
$\sum_{n=0}^{5} \frac{x^n}{n!}$	`n@0` Tab 5 Esc `(x^n)/n!`	
$\prod_{n=0}^{5} a_n$	`a_n` Esc `#n` Tab 0 Tab 5	Pi Products enclose arguments rather than multiply onto them

Other Expressions

The following examples show how to create various expressions, some of which are useful for working with graph theories.

To Create:	Type:	Comment
$a...b$	`a...b` or `a.b`	A single period between names is sufficient to indicate a range op
$-2...2$	`-2...2` or `-2..2`	Use at least two dots to create a range op between numbers; the first dot is interpreted as a decimal point
$x=a-2...a+2$	`x=a-2...a+2`	
$a=b$	`a=b`	
$x=a-2$	`x=a-2`	
$a<b$	`a<b`	
$a>b$	`a>b`	
$a\neq b$	`a Opt-= b`	Macintosh syntax
$a\neq b$	`a<>b`	Pascal syntax
$a\leq b$	`a < b`	Macintosh syntax
$a\geq b$	`a Opt-< b`	Pascal syntax
$\left[\frac{x^6}{6}\right]_{x=0}^{x=1}$	`x^6 Esc /6 Esc Opt-[` `x=0 Tab x=1`	
$\nabla=\left(\frac{\partial}{\partial x},\frac{\partial}{\partial y}\right)$	Opt-Shift-z `=(∂x,∂y)`	

Tips & Techniques

This chapter lists a few tips and techniques for using Theorist. This is probably not the best place to start if you are new to the program, but if you are familiar with the basic operations, these tips should help increase your speed and efficiency. The topics are grouped into three sections:

- Algebra
- Graphs
- Editing

Algebra

This section describes:

- Moving negations in equations
- Moving factors
- Making a real variable
- Making a variable a constant
- Enclosing an expression
- Using substitutions when factoring
- Factoring multivariate polynomials

Moving Negations

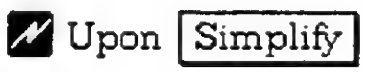
To move a negative sign from one side of an equation to the other, select the expression contained by the negative op (but not the negative op itself) and Isolate the expression to one side or the other.

Moving Factors

If a Move Over manipulation pulls over more than you want (perhaps a numerical coefficient), select the part that should not have moved, and use Move Over to drag it back.

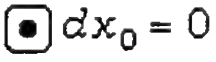
Making a Real Variable

There is no direct way to declare that a variable is real (as opposed to complex). However, you can create transformation rules that will do the trick:



Making a Variable a Constant

If x is of class Variable, any subscripted name that uses x as a base is also a variable. However, you can make a subscripted variable name act as a constant with a working statement:



Enclosing Expressions

To make a part of a sum or product act as a single unit, select the terms that should make up the subexpression and type an opening parenthesis, or simplify the expression in place.

Using Substitutions

Polynomials with non-simple terms (e.g., sin(*x*)) cannot be factored with the Factor manipulation. For example, the following expression does not factor:

$$(\sin[x])^3+3(\sin[x])^2+3\sin(x)+1$$

However, you can create a another expression, Substitute it into this expression and Factor the new expression:

$$\sin(x) = z$$

$$\square\ (\sin[x])^3+3(\sin[x])^2+3\sin(x)+1$$

$$\triangle\ (\sin[x])^3+3(\sin[x])^2+3\sin(x)+1 = z^3+3z^2+3z+1$$

$$\triangle\ (\sin[x])^3+3(\sin[x])^2+3\sin(x)+1 = (z+1)^3$$

Clever substitutions are also needed to factor an expression with pairs of terms. For example, the following expression does not factor in its current form:

$$(a-b)^3-6(-a+b)^2+12(a-b)-8$$

You could try to substitute an expression such as:

$$y = a-b$$

But this only substitutes into the two instances of $a - b$ (it will not substitute into $-a + b$). Instead, use an equation of the form:

$$a = b+y$$

Drag this equation over the icon for the first expression. This substitutes in throughout, to give you an expression in y:

$$y^3-6y^2+12y-8$$

And this expression responds favorably to Factor:

$$\triangle y^3-6y^2+12y-8=(y-2)^3$$

Factoring multivariate polynomials

Using the Factor manipulation on polynomials with many variables, or polynomials with symbolic coefficients, may not produce a satisfactory result. Sometimes the manipulation churns for a long time and accomplishes nothing.

In these situations, you can use Collect, Commute, and Simplify to step the polynomial to the form you want. For example, using Factor with the following polynomial produces a large and ugly result:

$$bx^2+x^2-ax-bx-abx-x+a+ab$$

The trick is to find the variable with the lowest, or one of the lowest, powers, and Collect on that variable. Commute the expression until b is the last factor of the first term:

$$x^2b+x^2-ax-bx-abx-x+a+ab$$

Then select the whole expression and Collect. This groups together terms that have b as a factor:

$$a+(a+x^2-ax-x)b+x^2-ax-x$$

The remaining terms are the same as the ones in the parentheses. Use Commute to rearrange them, then select these four terms and Simplify. This unifies them as a sum:

$$(a+x^2-ax-x)b+(a+x^2-ax-x)$$

Collect the whole expression to combine the two sums:

$$(a+x^2-ax-x)(b+1)$$

The polynomial in a can also be reduced. Use Collect to pull out the a, and then use Collect on the last two terms to pull out the x:

$$(-x+1)a+x^2-x$$

$$a(-x+1)+(x-1)x$$

Select both of these terms and use Collect again:

$$(a-x)(-x+1)$$

The complete "factored" result is:

$$([a-x][-x+1])(b+1)$$

This result can be cleaned up with Simplify:

$$(a-x)(b+1)(-x+1)$$

Note: These tricks do not work with all polynomials. Some just do not factor. Frequently, though, in algebraic derivations, you run into polynomials that can be factored, but it may take a little work with Commute, Collect, and Simplify.

Graphs

This section describes:

- Using subscripted variables
- Balancing graph bounds (around zero)
- Creating a rotating animation

Subscripted variables

If a subscripted variable does not appear in the dialog box for selecting the dependent and independent variables to use for a graph, enter a minimum or maximum expression using the subscripted name and make the expression a working statement. For example, if z_a does not appear in the dialog box, enter an equation, $z_a > 0$.

Subscripted names often refer to matrix elements and, if a subscripted name is interpreted that way by the program, matrix elements are not shown in this dialog box as possible variables.

Balancing Bounds

To create a range of values for the one of the graph bounds centered on zero, select the range (click on the three dots). Select In twice (Command-E), and enter a value. This only works if the first range value is already negative. You can also shift-click select more than one range, and change them all at once.

Rotating an Animation

To make a three-dimensional animation rotate, multiply the location vector of the plot you want to rotate by a rotation matrix. See the notebook "3D Coordinate Transformation" in the Graphics folder on your distribution disks for an example.

Editing

This section describes:

- Negating expressions
- Editing on the left
- Editing matrices
- Creating identity matrices

Negating Expressions

To negate an existing expression, select it and type Option-Hyphen. Just entering a minus sign op into an insertion point usually enters a minus sign and a new (empty) expression.

Editing on the Left

Many of the rules for keyboard entry work in reverse if you click on the left side of an expression. For example, if you click on the left side of a variable and type "+", a new question mark is added onto the left side rather than the right.

You can enclose a name or number in a square root by clicking on its left side and typing a backslash. This works for all prefix ops (square root, absolute value, partial derivative, negate, integral).

Editing Matrices

To select all the elements of a matrix, double-click-and-drag on any element. That is, click twice on one element, and hold and drag the mouse after the second click to any other element. To select a row (or column), click and drag across all of the elements in that row (or column). To select more than one row (or column) click on an element in the top-right corner and drag to the element in the bottom-left corner, or vice versa. With these selections you can Cut, Copy, or Delete the elements.

Removing a row (or column) leaves a blinking insertion point the length of the row (or height of the column). Pressing Delete removes the next row above (or the next column to the left). You can also create a blinking insertion point by clicking between matrix elements. Rows (or columns) can be pasted into this insertion point. Single characters,

or expressions copied from somewhere else, are pasted in as a set of elements, filling up the width of the row (or height of the column).

Creating Identity Matrices

An identity matrix contains a diagonal of ones, all other elements are zero, such as:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

To create an identity matrix, make a square matrix, select all elements individually (select the matrix as a whole then use Select In), choose **Row Number** from the **Input** menu, enter an equals sign, and choose **Column Number** from the **Input** menu. Then select the entire matrix and use the Calculate manipulation (Command-5).

Menus

Menu Commands

This chapter provides a list of all menu commands in Theorist with a brief description of what each option does. All menu items execute a command immediately, branch to sub-menus, or bring up a dialog box. Any menu option that ends in an ellipsis (...) brings up a dialog box. Options that end with a small arrow open sub-menus.

The Theorist menus are:

-
- File
- Edit
- Input
- Notebook
- Manipulate
- Graph
- Prefs

See Appendix A for a handy reference to Command-key equivalents.

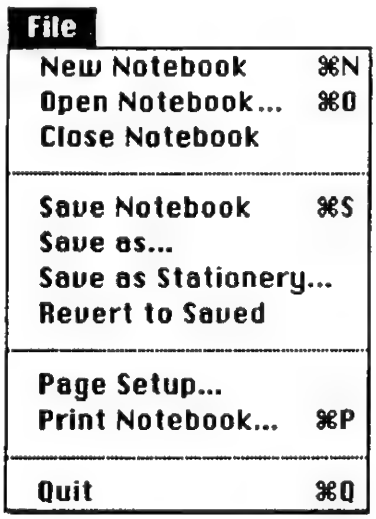
The Apple menu provides access to Desk Accessories (DAs), MultiFinder applications, and information about Theorist.

About Theorist

Shows the version number and identification information about Theorist, including copyright notices and other important messages.

File

The File menu provides commands for creating, opening, closing, saving, and printing notebooks and stationery files, and quitting Theorist.



New Notebook Creates a new notebook (named "Untitled 1," "Untitled 2," etc.) with the default propositions. All preferences are taken from the current notebook, if any, or are otherwise set to defaults.

Open Notebook... Opens an existing notebook file or stationery file.

Close Notebook Closes the current notebook window, as if you clicked on the close box. If there are any unsaved changes, you are asked whether you want to save the file.

Save Notebook Saves the current notebook window in its file. If it is a new untitled notebook (i.e., it has not yet been saved), you are prompted for a file name.

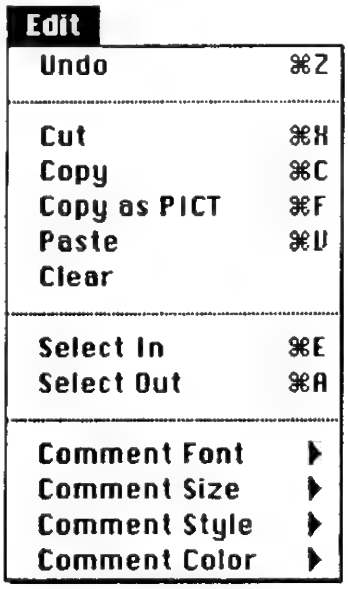
Save as... Prompts you for a new file name (and folder or volume) to use to save the current notebook. Any file previously associated with the current notebook remains as most recently saved.

Save as Stationery... Prompts you for a file name (and folder or volume) to save the current notebook as a stationery file.

Revert to Saved	Disposes of all changes to the current notebook since it was last saved or opened. You are asked to confirm this command.
Page Setup...	Brings up a dialog box for you to change or verify the print settings. The dialog varies depending upon the type of printer you are using.
Print Notebook...	Prints the current notebook. A dialog box appears with printing options. The dialog varies depending upon the type of printer you are using.
Quit	Quits Theorist, prompting you to save changed files, if any. Any startup notebooks (named "Notebook 1," "Notebook 2," etc.) that opened automatically, are saved automatically.

Edit

The **Edit** menu provides commands for editing equations, expressions, comments, propositions, and graphs.



Undo Reverts the current notebook to its state before the most recent change. Some changes cannot be undone.

Cut Copies the current selection to the clipboard, then deletes the selection. Expressions are stored as text in the Fortranish format.

Copy Copies the current selection to the clipboard. Expressions are stored as text in the Fortranish format.

Copy as PICT Copies the current selection (expression or graphic) to the clipboard. Selected expressions are stored as bit-mapped images. Mathematical expressions include embedded Expressionist hints. Graphs are copied as graphic images as specified in the Graph preferences (on the Prefs menu).

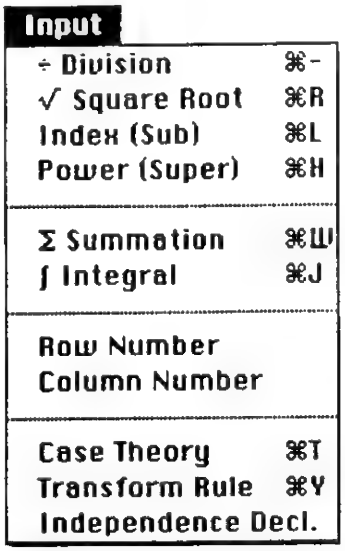
Paste Pastes the contents of the clipboard replacing, or just after, the current selection. If there is no selection, the clipboard contents is pasted at the bottom of the current notebook.

Clear Deletes the current selection; same as pressing Delete on the keyboard.

Select In	Changes each current selection to a multiple selection, selecting each subexpression of the current selection. Hold down Shift to select only the first subexpression.
Select Out	Changes each current selection to a selection of the enclosing expression or proposition. If two or more selections collide, they undergo pairwise annihilation; that is, if an even number of selections collide, no selections are left, but if an odd number of selections collide, one selection remains.
Comment Font, Size, Style, Color	Use the commands on these sub-menus to change the font, size, style, and color of selected text in a text comment proposition.

Input

The **Input** menu provides commands for inserting propositions and parts of expressions.



Division

Inserts the division op into the current expression(s). This is the same as clicking on the x/y palette button or typing "/".

Square Root

Inserts the square root op into the current expression(s). This is the same as clicking on the $\sqrt{x}$ palette button or typing "\".

Index (Sub)

Inserts the index, or subscript, op into the current expression(s). This is the same as clicking on the x_y palette button or typing "_".

Power (Super)

Inserts the power, or superscript, op into the current expression(s). This is the same as clicking on the x^y palette button or typing "^".

Summation

Inserts the summation op into the current expression(s). This is the same as typing "@". Clicking on the Σx palette button inserts a summation op followed by a question mark.

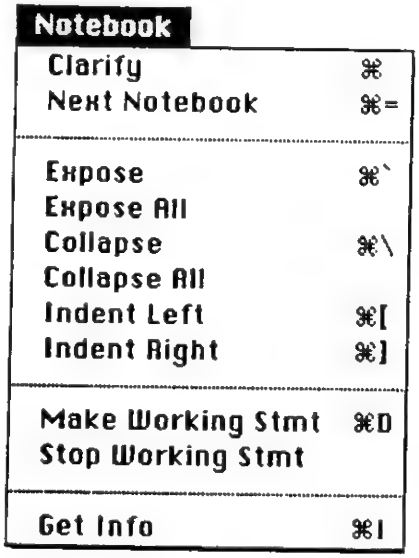
Integral

Inserts an integral with limits into the current expression(s). This is the same as clicking the $\int_y^z x$ palette button or typing Option-4.

Row Number Column Number	Inserts an integer into the current selection (if the selection is some or all of the elements of a matrix) that is the row or column number of the matrix element. If a selection is inside nested matrices, the outermost matrix is used to determine the row or column number.
Case Theory	Creates a new Case Theory that encloses all currently selected propositions (if any) and all daughter or dependent propositions (if any).
Transform Rule	Inserts one or more new transformation rule proposition(s). If one or more equation is selected, a transformation rule is created for each equation. The left-hand side of each equation becomes a pattern and the right-hand side becomes a replacement.
Independence Decl.	Inserts a new Independence Declaration after the current selection or at the bottom of the current notebook if there is no selection.

Notebook

The **Notebook** menu provides various commands for arranging and working with propositions and notebooks.



Clarify — Scans the current notebook for any syntax errors, undeclared names, or other problems. It also regenerates all graphs from current expressions.

Next Notebook — Makes a different notebook window the current notebook, if you have more than one notebook open. Repeated commands cycle through all open notebooks.

Expose — Un-collapses the selected item(s). If a proposition, it exposes all daughter propositions; if an expression collapsar, it exposes it for view.

Expose All — Un-collapses all propositions and all collapsars in the current notebook.

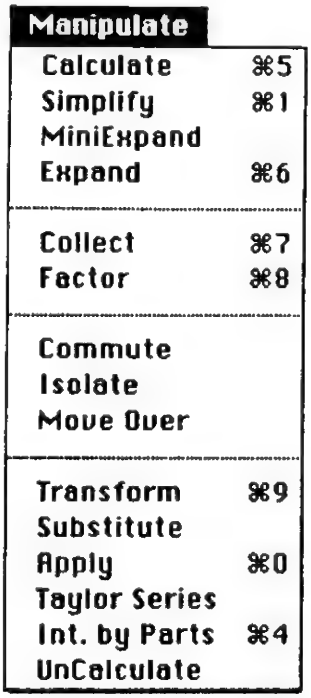
Collapse — Collapses the selected item(s). If a proposition, it collapses all daughter propositions beneath that proposition; if an expression, it collapses it into a collapsar.

Collapse All — Collapses all propositions in the current notebook. Collapse All does *not* collapse any expressions.

Indent Left	Moves each selected proposition one level to the left. If a selected proposition is an inline proposition, it becomes a daughter proposition. The selections can be all or part of a proposition or expression.
Indent Right	Moves each selected proposition one level to the right. If a selected proposition is already a daughter proposition, it becomes an inline proposition. The selections can be all or part of a proposition or expression.
Make Working Stmt	Turns the currently selected assumption(s) or conclusion(s) into working statements. Any existing working statements that conflict with the new one(s) stop being working statements. The selections can be all or part of a proposition or expression.
Stop Working Stmt	Changes selected working statement(s) to regular statements. If the statement is required to be a working statement (e.g., for a graph), the statement (or one like it) becomes a working statement again immediately. The selections can be all or part of a proposition or expression.
Get Info	Displays a dialog box for each selected object that describes what the object is and what its relationship is to other objects. You can get information about any proposition, name, number, expression, or graph theory.

Manipulate

The **Manipulate** menu provides commands for evaluating, rearranging, and exploring expressions.



Calculate Evaluates an expression to a number like a pocket calculator.

Simplify Executes a wide range of operations designed to reduce an expression to a simple canonical form.

MiniExpand Executes the Expand manipulation on the outer-most layer of an expression.

Expand Executes a range of operations designed to expand an expression to its most explicit form.

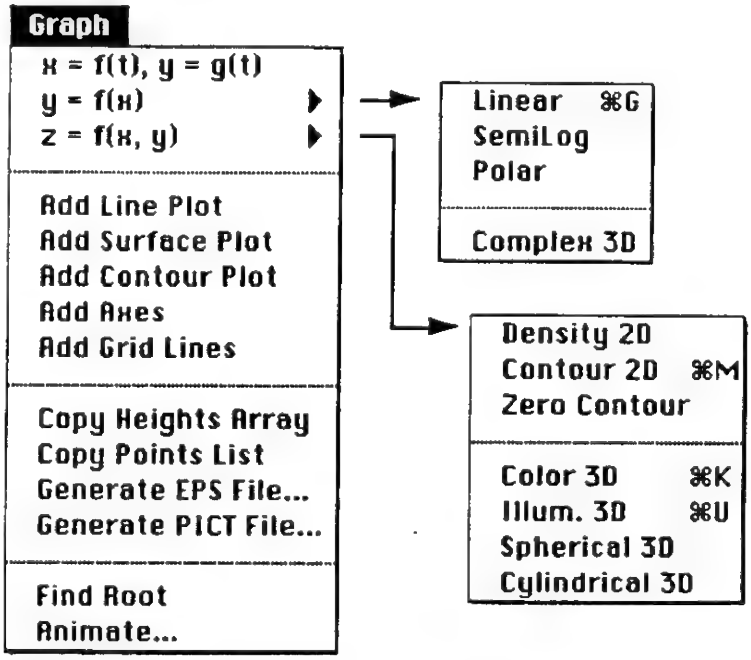
Collect Executes several operations designed to group similar parts of an expression together.

Factor Executes several operations designed to dismantle an expression into its most fundamental parts or terms.

Commute	Rearranges terms in a sum, or factors in a product or dot product.
Isolate	"Solves" an equation for a selected expression.
Move Over	Moves an expression from one side of an equation to the other.
Transform	Executes all transformation rules set to execute "Upon Transform." Transform is designed as a means for you to extend Theorist's capabilities by identifying expressions of particular forms and turning them into different forms.
Substitute	Replaces expressions of a particular form with an equivalent expression of a different form.
Apply	Selects each side of a selected equation (or the numerator and denominator of a selected fraction) so that all subsequent manipulations are applied equally to both sides. Can also augment expressions in many other ways with out destroying an expressions logical integrity.
Taylor Series	Generates a Taylor series of a continuous function.
Int. by Parts	Integrates the selected part of a larger integral.
UnCalculate	Undoes the effects of the Calculate manipulation (if possible), eliminating round-off error and symbolically reconstructing numeric values.

Graph

The **Graph** menu provides commands for making and modifying graphs, and for extracting data from graphs.



x = f(t), y = g(t)

Creates a 2-D parametric plot, from the selected expression, with two dependent variables being functions of one independent variable, the parameter.

y = f(x)

Each command on this sub-menu creates a graph with one dependent and one independent variable.

Linear

Creates a rectangular 2-D graph with a linear coordinate system.

SemiLog

Creates a rectangular 2-D graph with linear horizontal coordinates and logarithmic vertical coordinates.

Polar

Creates a rectangular 2-D graph with a polar coordinate system.

Complex 3D

Creates a rectangular 3-D graph containing a colored Surface plot depicting a complex function on the complex plane.

z = f(x, y)

Each command on this sub-menu creates a graph with one dependent variable and two independent variables.

Density 2D	Creates a rectangular 2-D graph displaying the dependent variable as a range of gray values.
Contour 2D	Creates a rectangular 2-D graph displaying the dependent variable as a selection of labeled contours.
Zero Contour	Creates a rectangular 2-D graph displaying a single contour line where the dependent variable equals zero.
Color 3D	Creates a rectangular 3-D graph containing a colored Surface plot.
Illum. 3D	Creates a rectangular 3-D graph containing an illuminated Surface plot.
Spherical 3D	Creates a 3-D graph with a spherical coordinate system containing an illuminated Surface plot.
Cylindrical 3D	Creates a 3-D graph with a cylindrical coordinate system containing an illuminated Surface plot.
Add Line Plot	Adds a Line plot to an existing graph theory.
Add Surface Plot	Adds an illuminated Surface plot to an existing graph theory.
Add Contour Plot	Adds a Contour plot to an existing graph theory.
Add Axes	Adds a set of Axes to an existing graph theory.
Add Grid Lines	Adds a set of Grid lines to an existing graph theory.
Copy Heights Array	Creates a tab and return delimited text list of data point values from a Line or Surface plot and places the list in the clipboard.
Copy Points List	Creates a tab and return delimited text list of data point values from a Line or Surface plot and places the list in the clipboard.
Generate EPS File...	Creates an Encapsulated PostScript file (EPSF) from a graph. Hold down the Option key to generate an Encapsulated PostScript text file (TEXT) without attached QuickDraw picture.

Generate PICT File...

Creates a PICT file from a graph. The picture format is identical to that created with **Copy as PICT** on the **Edit** menu, but rather than putting the image in the clipboard, this command creates a named file.

Find Root

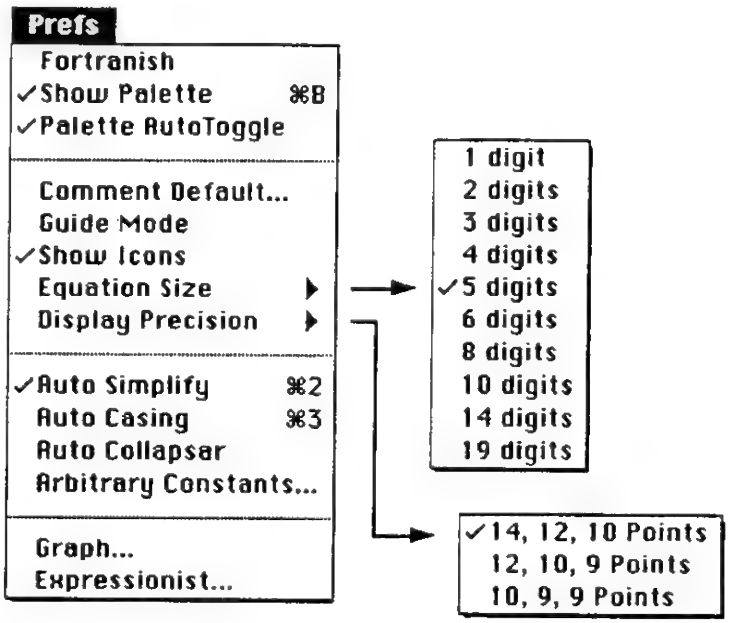
Finds the numerical value of the horizontal coordinate that zeros the vertical coordinate of a Line plot in a 2-D graph theory. Also finds the numerical value of the horizontal and vertical coordinates that simultaneously zeros two Contour plots in a 2-D graph theory. In both cases, values for all variables are presented in a separate case theory.

Animate...

Brings up the animation dialog to animate a graph. Select an animation variable first.

Prefs

The **Prefs** menu provides commands and options for customizing notebook display and printing. Each setting is global to a notebook and is stored with each notebook file. Certain items use a check mark next to their name in the menu to indicate that they are "on."



Fortranish — Determines whether keyboard input follows FORTRAN precedence rules explicitly. (Default: off)

Show Palette — Shows or hides the entry palette. (Default: on)

Palette AutoToggle — If this option is on, the palette flips between the function side and the variable side, anticipating your next click. If off, the palette toggles only when you click on its icon. (Default: on)

Comment Default... — Brings up a dialog box to set the font and size for new comments. Previously created comments are not affected. To edit existing comments, use the Edit menu. (Default: 12 point Times)

Guide Mode — Displays all expressions with as many parentheses as possible. (Default: off)

Show Icons — Shows or hides all proposition icons and graph buttons. (Default: on)

Equation Size	Sets the font sizes of displayed expressions. Each option sets the main equation size, the size for superscripts (and subscripts), and the size for super-superscripts (and sub-subscripts). (Default: 14, 12, and 10 points)
Display Precision	Sets the total number of decimal digits, counting from the leading digit, shown on the screen for numerical values. This preference affects display only; the full 19 digit precision is maintained internally. (Default: 5 digits)
Auto Simplify	Executes the Simplify manipulation (up to ten times or until the expression stops changing) after executing any other manipulation other than Apply and Commute. (Default: on)
Auto Casing	Forces manipulations to generate separate case theories or arbitrary constants to display all results of a manipulation that generates multiple values. (Default: off)
Auto Collapsar	Displays all expressions longer than the screen width as collapsars. (Default: off)
Arbitrary Constants...	Brings up a dialog box to change the settings for arbitrary constants.
Graph...	Brings up a dialog box for setting various options to control graph display, printing, and exporting.
Expressionist	Brings up a dialog box to specify various parameters for equations to be exported from Theorist to Expressionist.

Appendices

Appendix A: Keyboard Maps

This appendix contains:

- A table of Greek characters
- A figure of Command key equivalents for menu commands
- Keyboard maps

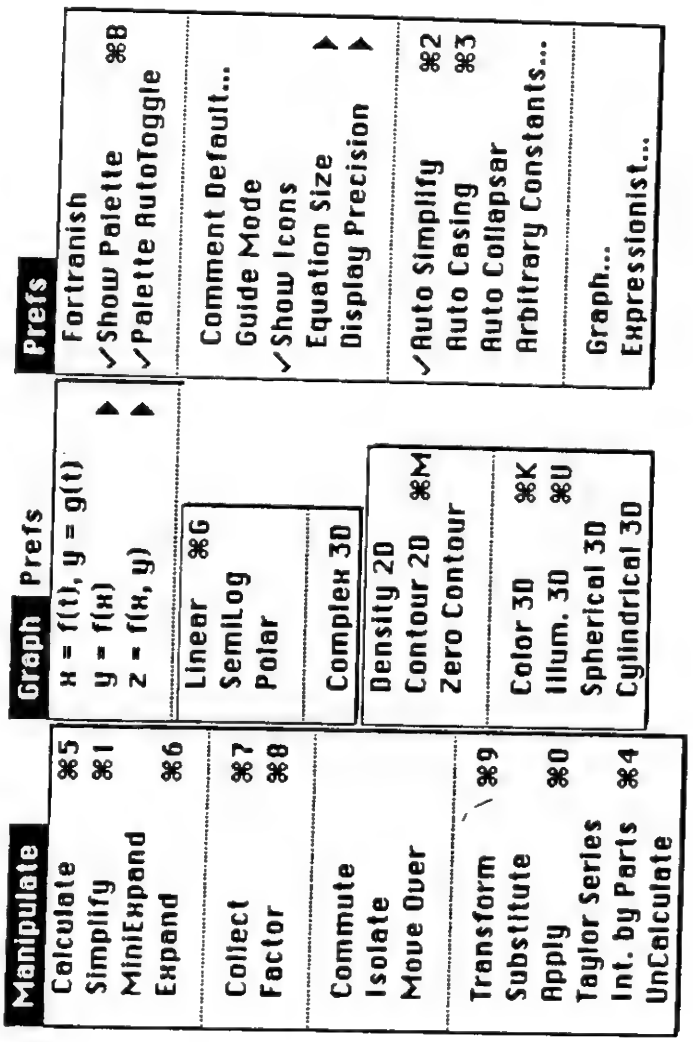
The following table lists the keystrokes to use for upper- and lowercase Greek characters.

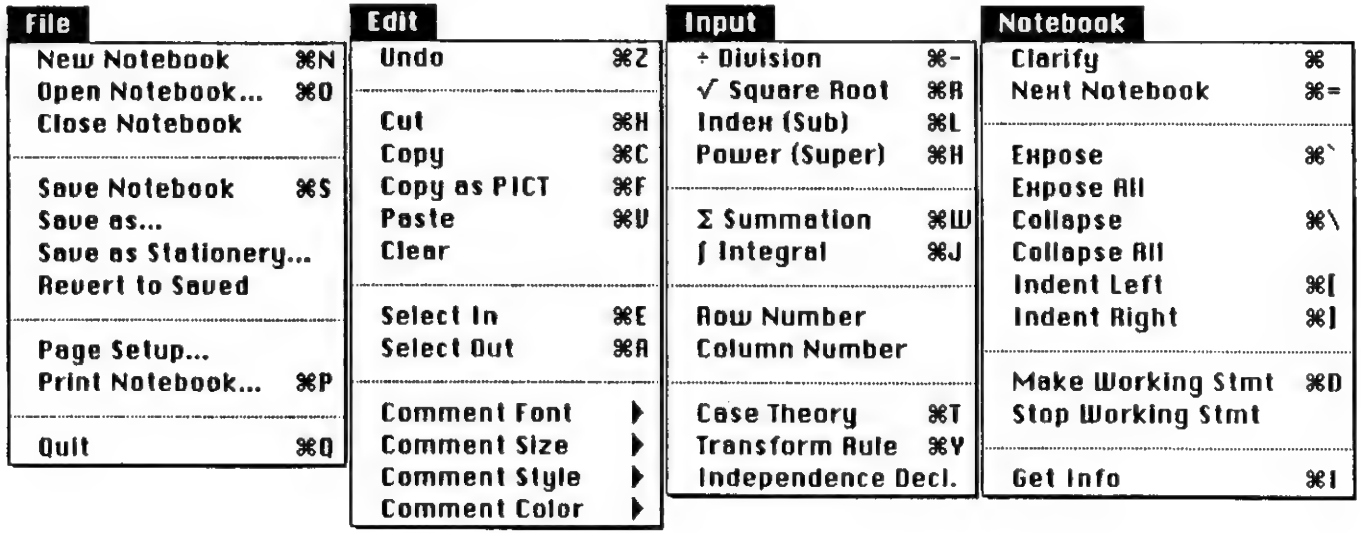
Keystroke:	a	b	c	d	e	f	g	h	i	j	k	l	m
Option:	α	β	χ	δ	–	ϕ	γ	η	–	φ	κ	λ	μ
Opt-Shift:	ℵ	⟨	–	Δ	ε	Φ	Γ	ℏ	ℑ	ϑ	⟩	Λ	–

Keystroke:	n	o	p	q	r	s	t	u	v	w	x	y	z
Option:	–	–	π	θ	ρ	σ	τ	–	υ	ω	ξ	ψ	ζ
Opt-Shift:	ν	–	Π	Θ	ℜ	Σ	–	–	ϒ	Ω	Ξ	Ψ	∇

If you want to learn the Command key equivalents for Theorist's menu options, you can copy the following two figures onto card stock and tape them to the top of your monitor.

Note, the Graphs menu has been modified for this figure.




File
New Notebook ⌘N
Open Notebook... ⌘O
Close Notebook
Save Notebook ⌘S
Save as...
Save as Stationery...
Revert to Saved
Page Setup...
Print Notebook... ⌘P
Quit ⌘Q
Edit
Undo ⌘Z
Cut ⌘H
Copy ⌘C
Copy as PICT ⌘F
Paste ⌘V
Clear
Select In ⌘E
Select Out ⌘A
Comment Font
Comment Size
Comment Style
Comment Color
Input
÷ Division ⌘-
√ Square Root ⌘R
Index (Sub) ⌘L
Power (Super) ⌘H
Σ Summation ⌘W
∫ Integral ⌘J
Row Number
Column Number
Case Theory ⌘T
Transform Rule ⌘Y
Independence Decl.
Notebook
Clarify ⌘
Next Notebook ⌘=
Expose ⌘`
Expose All
Collapse ⌘\
Collapse All
Indent Left ⌘[
Indent Right ⌘]
Make Working Stmt ⌘D
Stop Working Stmt
Get Info ⌘I

These maps show a picture of the standard Macintosh keyboard and, on each key, the effect of pressing that key. If a key is blank, nothing happens when you press it. Maps are provided for the keyboard with:

- No modifiers
- Shift
- Option
- Option and Shift
- Command

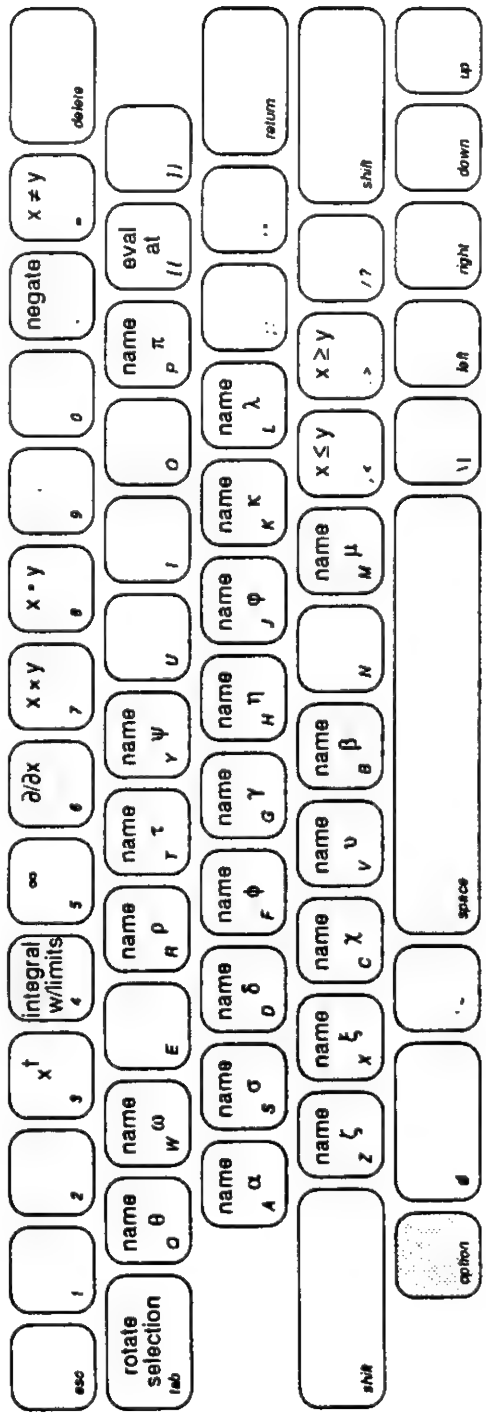
Where "name" appears on a key, that character is entered as a part of a name.

No Modifiers

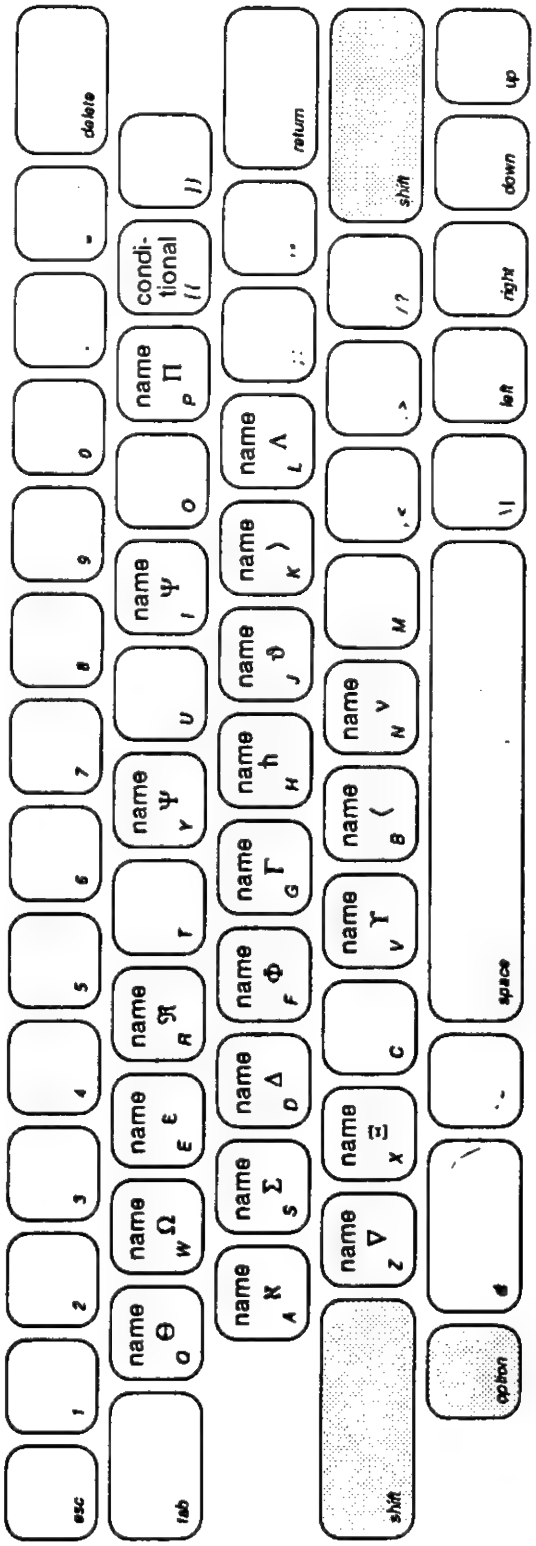
Shift

select escape esc
x!
Σ x
Π x
∫ x
x^y
xy
open paren 9
close paren 0
x_y
x + y
delete selection delete
rotate selection tab
name Q
name W
name E
name R
name T
name Y
name U
name I
name O
name P
open paren [
close paren]
name A
name S
name D
name F
name G
name H
name J
name K
name L
new line in comment return
shift
name Z
name X
name C
name V
name B
name N
name M
x < y
x > y
wild card /?
shift
option
space
abs value \|
left
right
down
up

Option


esc
1
2
x†
3
integral w/limits
4
∞
5
∂/∂x
6
x × y
7
x • y
8
9
0
negate
x ≠ y
delete
rotate selection
tab
name
Q θ
name
W ω
E
name
R ρ
name
T τ
name
Y ψ
U
I
O
name
P π
eval at
{{
}}
name
A α
name
S σ
name
D δ
name
F φ
name
G γ
name
H η
name
J φ
name
K κ
name
L λ
return
shift
name
Z ζ
name
X ξ
name
C χ
name
V υ
name
B β
N
name
M μ
x ≤ y
x ≥ y
/?
shift
option
space
left
right
down
up

Option-Shift



Command

esc
Simp-lify
1
2
3
Int. by parts
4
Calcu-late
5
Expand
6
Collect
7
Factor
8
Trans-form
9
Apply
0
x / y
-
delete
tab
Quit
Q
Σ x
W
Select In
E
square root
R
Case Theory
T
Y
Illum. Graph
U
Get Info
I
Open Notebk
O
Print
P
Indent Left
{{
Indent Right
}}
Select Out
A
Save Notebk
S
D
Copy as Pict
F
Graph
G
H
New Notebk
N
Color Graph
K
x y
L
return
shift
Undo
Z
Cut
X
Copy
C
Paste
V
B
M
shift
option
Expose
.
space
Clarify
Col-lapse
\|
left
right
down
up

Appendix B: Using Projectionist

Projectionist is a basic program for showing animation files created in the PICS format. This program is included on your distribution disks. Two example animations are also included. This appendix ends with a brief description of how each file was created.

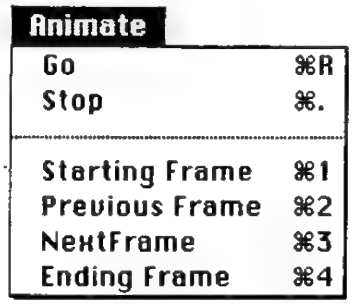
To launch Projectionist, double click on its icon. Once the program is running, you can open any saved animation file to view it. More than one file can be open and running at the same time. See the animation section in the Graphs chapter for a description of creating animation files. The program offers four menus of commands:

- File
- Edit
- Animate
- Speed

To open an animation file, choose **Open** from the **File** menu. A simple window is displayed, the animation file is loaded into memory, and the show begins. The size of the window and all graph settings (including color and resolution) are determined when you create the animation and cannot be altered using Projectionist. To close the file, click on the close box in the upper left-hand corner or choose **Close** from the **File** menu.

From the Animate menu you can start and stop the animation film. You can also step through the sequence of frames one at a time (forward or backwards), and jump to the first or last frame. Each of the single-frame commands stop a running animation. Command key equivalents are available for each of these commands.

Use the Speed menu to set the number of frames shown each second.



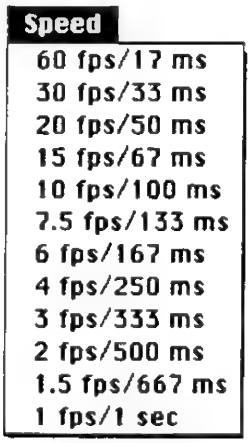
The actual number of frames per second (fps) depends on:

- The speed of your Macintosh
- The size (number of pixels) of each image
- The depth (number of bits per pixel) of each image
- How much memory is available

Animation is mostly a process of moving data. The faster your Macintosh can move the image data from memory to screen, the faster the animation can run. In some situations, your machine may not be able to display animation frames faster than, say, fifteen frames a second. Setting the speed to a faster setting will not change the display.

If Projectionist is not allocated enough memory to load the entire file into RAM, some frames are read from disk. This produces a much slower animation. To change the memory allocation, select the application icon (before launching the program), and choose **Get Info** from the **File** menu. Replace the number in the box in the lower right-hand corner of the dialog box with a larger number.

The Speed menu lists the number of frames per second and the approximate length of time each frame is displayed in milliseconds:



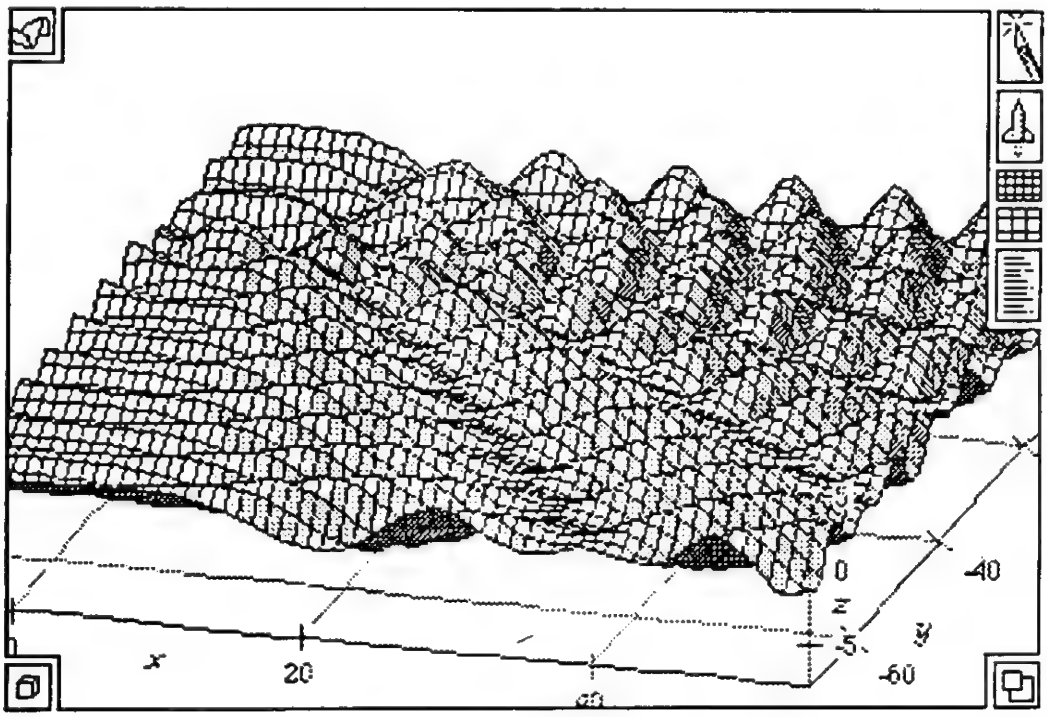
The PICS file format was developed by Macromind Corporation as a standard file format to store animated images. There are many animation programs that import and export PICS files including several public domain programs. The standard format consists of a resource file with one PICT (or QuickDraw) resource for each image. For more information, see *Inside Macintosh*, Volume 1. Contact Macromind for the PICS file format specifications.

Animation Files

Your distribution disks contain two animation files, one in full color, one in black and white. This section describes the steps used to create these files. To view either file, open it from inside Projectionist or any other PICS display program.

Electron Diffraction

This file is a color animation of the wave function of an electron diffracted through two slits. You can see the nodes of the wave as diagonal lines in black (black designates zero) that radiate from the origin. The following figure displays a portion of this radiation pattern. The animation is created from a close up of a small fraction of this graph.



Use the following steps to recreate the Electron Diffraction animation.

Enter an equation for the static wave function:

$$\psi = e^{i\sqrt{x^2+(y+a)^2}} + e^{i\sqrt{x^2+(y-a)^2}}$$

Enter another equation for the spin of time:

$$z = \mathrm{Re}\left(\psi e^{-it}\right)$$

Add an animation variable to the static wave function, and an equation that temporarily sets the animation variable to 20:

$$\square\ \psi = e^{i\sqrt{x^2+(y+a)^2}} + e^{i\sqrt{x^2+(y-a)^2}}$$
$$\square\ a = 20$$

And set the variable t equal to one:

$$\square\ t = 1$$

Select the spin of time equation and create an Illuminated 3D graph. Rather than accept the default dependence relations, declare x and y to be the dependent variables (keep z as the independent variable). All four equations are turned into working statements.

Open the graph details and change the color scheme of the Surface plot from Solid to Complex, the shade expression from z to ψ, and turn the mesh off:

Surface at (x,y,z) where x = west ... east and y = south ... north ;
Illuminated surface has no mesh and is shaded using Complex coloring;
ψ = bottom ... top determines brightness and hue (from magnitude and phase).

Adjust the viewport controls to: True Proportions, cropped Tightly, as seen through an Infinitely distant lens. Change the graph bounds to the following values (you can do this directly by editing the numbers, or approximately by using the Rocket and Knife):

$17 \ldots 32$ = west...east
$-45 \ldots -27$ = south...north
$-2 \ldots 2$ = bottom...top

Select either of the animation variables in the static wave function (a), and animate the graph. Depending on the rotation of the graph, the resolution, and the number of frames used for the animation, your animation should be similar if not identical to the Electron Diffraction file.

Pulse Vibration

This file is a black and white animation of the response of a mechanical system to a pulse. The panel's vibration displays simple damped oscillation of a surface.

Enter an equation describing the displacement of the surface over time, and a place holding equation for the animation variable (t):

$z = y\sin(x)\cos(3t)e^{-t}$

$t = 6$

The first portion of the equation, $y = \sin(x)$, describes the shape of the deflection. The second portion, $\cos(3t)$, describes the frequency of the oscillations, and the last part, e^{-t}, defines the rate of decay.

Select the first equation and create an Illuminated 3D graph. Open the graph details and adjust the graph bounds to:

$-3 \ldots 3$ = west...east

$-0 \ldots 3$ = south...north

$-1 \ldots 1$ = bottom...top

Then select both occurrences of the animation variable (t), and animate the graph.

Appendix C: Using Expressionist with Theorist

When you use **Copy as PICT** on the **Edit** menu to Copy an equation or expression, Theorist generates a QuickDraw image of the current selection. The picture generated is a bitmapped image of what appears on screen. Although not of high resolution, this image can be moved to almost any Macintosh program, and is satisfactory for screen display or for informal printed presentations.

The picture created by Copy as PICT also includes Expressionist comments. You can Paste the figure into Expressionist for high quality equation typesetting. You can edit the equation in Expressionist, just as though you had typed it in directly. You can then Copy and Paste the image into any Macintosh desktop publishing program for a typeset mathematical expression.

You can also Copy equations from Expressionist into Theorist. An Expressionist preferences file is supplied on your distribution disks to translate Expressionist equations into Theorist equations. To use it, you must own Expressionist. Move the file called "Expr ->Theo Prefs" into your System folder and rename it "Expressionist Prefs".

To Copy an Expressionist equation into Theorist, select an expression in Expressionist, then choose **Copy as Text** (Command-F) from the **Edit** or **Expressionist** menu, click on the Theorist window and Paste it in.

The translation is not exact because Expressionist equations use very few syntax rules. Some editing may be necessary. In particular, names that should be multiplied may be concatenated, creating unintended new names. For example, if the two names *x* and *b* are next to each other in an expression in Expressionist (with out a separating space), when brought into Theorist these two names become the single name "bx". To correct this, click between the two characters and press space.

Appendix D: Distribution List

The following files and folders are included on the floppy disks you received when you purchased the program.

Theorist 1.1
The main program. This version runs on all Macintosh models.

Theorist 1.1 Mac II
The main program modified to take advantage of the 68020 and 68030 CPUs and the 68881 and 68882 math co-processors. Use this version if you work on a Mac II, Mac IIx, Mac IIcx, Mac IIci, Mac IIfx, Mac SE/30 or any Macintosh equipped with a 68020, 68030, or compatible CPU, *and* a 68881, 68882 or compatible floating point co-processor chip.

Graphics Folder
A folder containing several notebooks with sample graphs demonstrating some of the graphs you can create.

Color Graphics Folder
A folder containing several notebooks with sample color graphs demonstrating some of the graphs you can create.

Animation Folder
A folder containing several notebooks, each of which contains a graph suitable for animation.

Projectionist Folder
This folder contains Projectionist, a basic animation playback utility, and two demonstration animation files created with Theorist.

Expressionist Demo Folder
A folder containing a demonstration version of Expressionist, the Macintosh equation editor from Prescience, and a manual for the demo that you can open from inside any word processor.

Expr ->Theo Prefs
A preferences file for Expressionist. Put this file in your System folder and rename it "Expressionist Prefs". This file provide the information Theorist requires to import an expression from Expressionist.

Mathematics Folder
A folder containing several notebooks of mathematical rules including the following folders and files:

- Finite & Infinite Series
  Rules for summations

- Integral Tables
  Folder of notebooks containing integration rules

- Laplace Transforms
  Rules for Laplace and Inverse Laplace transforms

- 3D Calculus
  Rules for three-dimensional linear algebra and vector calculus

- Standard Rules & Declarations
  Rules for transcendental functions, and many other common rules

- Number Theory
  Rules for working with integers

- Transcendental Functions
  Folder of notebooks containing rules for trigonometric, logarithmic, power, and hyperbolic functions

- Special Functions
  Folder of notebooks containing numerical definitions of special functions including Bessel functions, orthogonal polynomials, and Spherical Harmonics

- Re( ) & Im( )
  Rules for real and imaginary functions

- Statistics
  Rules for calculating means, standard deviations and linear regression over matrix numbers

- Units
  Rules defining a wide variety of units

Glossary

This glossary defines some general mathematical terms, and terms specific to Theorist.

Adjoint
The transpose of the conjugate of a matrix, or the conjugate of a scalar.

Algorithm
A mechanical procedure for solving a problem in a finite number of steps.

Animation
A sequence of images displayed in quick succession to simulate movement.

Apply
A Theorist manipulation whereby two parts of an equation or expression can be augmented simultaneously so that the two parts cancel out.

Arbitrary constant
A constant included as part of a result when multiple solutions are possible, each of which is described by substituting any value for the arbitrary constant. Also called the constant of integration.

Arbitrary integer
An integer constant included as part of a result when multiple solutions are possible, each of which is described by substituting any integer value for the arbitrary constant.

Argument
An expression passed to a function or op, sometimes enclosed in parentheses.

Associative
A property of a binary operator. When three or more elements are combined with a binary operator, that is associative for those elements, the result does not depend on how the elements are grouped into pairs. If multiplication is associative across the names *A*, *B* and *C* then *A*(*BC*) = (*AB*)*C*.

Auto Casing
A Theorist option. If Auto Casing is on, manipulations that generate more than one possible solution create case theories or arbitrary constants.

Auto Collapsar
A Theorist option. If Auto Collapsar is on, large expressions generated by manipulations are displayed in abbreviated form.

Auto Simplify
A Theorist option. If Auto Simplify is on, most manipulations conclude by executing the Simplify command.

Axis
A Theorist plot proposition that displays a labeled axis (with tick marks) in a graph viewport.

Base
A name, number, or expression that is raised to a power (with a superscript) or qualified with an index (or subscript).

Behavior
Properties of names. Theorist uses fifty seven predefined behaviors; you can also define and declare the behavior of a names with an equation.

Boolean
A value that is True or False (or 1 or 0).

Case theory
A Theorist proposition. Case theories enclose side derivations. What ever is true in enclosing theories is true in the contained case theory; what ever is true inside a case theory is ignored outside of that theory.

Ceiling
A Theorist predefined name that acts as a function. It rounds a given argument to the next highest integer. See floor.

Class
The Theorist categorization system of names and expressions that determine how they behave algebraically. There are five classes: Constant, Variable, M-Linear Operator, D-Linear Operator, Function.

CMY
A color encoding system that describes colors by their Cyan, Magenta, and Yellow components.

Collapsar
A Theorist expression that has been abbreviated so that some terms or factors are replaced by an ellipse (...).

Collapse
A Theorist command to abbreviate long expressions (into collapsars) or conceal the daughter propositions of a proposition.

Collapse All
A Theorist command to collapse all propositions in the current notebook.

Collect
A Theorist manipulation that collects common factors out of a sum or product and other similar tasks.

Commutative
A property of a binary operator. When two or more elements are combined with a binary operator that is commutative for those elements, the result does not depend on the order of the elements. If multiplication is commutative over the names *A* and *B* then $AB = BA$.

Commute
A Theorist manipulation used to interchange two expressions in a product or sum, or reverse the order of all (or some) of the elements in a sum or product.

Complex arithmetic
Arithmetic of complex numbers.

Complex numbers
Numbers that have real and imaginary components. The imaginary component is designated with the constant *i*, the square root of negative one (e.g., $a + bi$).

Complex plane
A plane with pair of orthogonal axes such that one axis corresponds to the real part of a complex number (a) and the other to the imaginary part (b). Also called an Argand diagram.

Conjugate
The conjugate of a complex number a+b*i* is the complex number a–b*i*. These two form a conjugate pair and each is the conjugate of the other. The complex conjugate of a matrix is formed by replacing each element of the matrix with its conjugate.

Constant
A Theorist class. Names of class constant have fixed numerical values that do not change, unless you change them. Graph bounds (top, bottom, etc.) are constants; using the rocket or knife changes them.

Contour plot
A Theorist plot proposition. Contour plots represent a surface with a series of labeled elevation lines. Contour plots are drawn in panels like Surface plots.

Copy as PICT
A Theorist command that copies the current selection to the Macintosh clipboard as a picture (using the PICT format).

Cross product
A Theorist op. A cross product is the product of two vectors with either two or three elements. For two-vectors, the result is a scalar; for three-vectors, the result is another vector. If *A* and *B* are three vectors and *C* is the cross product of *A* and *B*, the length of *C* is the product of the lengths of *A* and *B*, and the sine of the angle between the two vectors.

D-Linear Operator
A Theorist class. Names of class D-Linear Operator are derivatives or other ops that are neither commutative nor associative under multiplication.

Dependent variable
A variable in an equation whose value is contingent on the independent variables in that equation, and the equation itself.

Derivative
The rate of change of a dependent variable (or function) with respect to an independent variable. Theorist works derivatives symbolically, and calculates them numerically.

Details
Part of a Theorist graph. The details box contains a description of each of the graphic elements (plots) shown in the graph.

Determinant
A sum of products of matrix elements in a particular pattern (some terms are negative). The determinant of $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ is $ad - bc$.

Differential
A product of the differential operator d with any variable. Its value is infinitesimal; you get derivatives by dividing different differentials.

Differentiation
The process of finding the derivative of a function. Differentiation is the inverse process of integration.

Display precision
A Theorist option that determines how many digits are displayed for numeric values.

Dot product
A Theorist op. A dot product is the scalar product of two vectors (with either two or three elements). If C is the dot product of A and B, the length of C is the product of the lengths of A and B, and the cosine of the angle between the two vectors.

Entry palette
Part of Theorist's main display. Use the palette to enter names and functions. The palette consists of the variable palette (for entering variables and constants) and the function palette (for entering ops and functions).

EPS
Acronym for Encapsulated PostScript. EPS files (type EPSF) use the PostScript language to describe computer generated images, and are used to move images between programs and systems.

Equation
A Theorist expression consisting of two subexpressions connected by the equals op.

Equation size
A Theorist option that determines the set of font and icon sizes to use for displaying expressions.

Expand
A Theorist manipulation used to break up an expression into a sum of smaller pieces, and other similar tasks.

Exponent
A number or expression placed as a superscript to a base indicating repeated multiplication.

Expression
A Theorist entity. An expression is a name, a number, or an op that encloses one or more name or number.

Factor
A Theorist manipulation used to consolidate polynomials into factored form, and other similar tasks.

Factorial
A Theorist op. The factorial of an integer is the number obtained by multiplying all the positive integers less than or equal to the given integer. The factorial of other real and complex numbers is defined by using the Gamma function.

Floor
A Theorist predefined name that behaves as a function. It rounds a given argument to the next highest integer. See ceiling.

Fortranish
A Theorist option. If Fortranish is on, expressions are entered from the keyboard in accordance with the precedence rules for operators established by FORTRAN and other programming languages.

Function
A Theorist class. Names of class function use the function op to enclose one or more input arguments, and they return a value determined by the definition of the particular function.

Function palette
One side of the Theorist entry palette, used to enter ops and functions. This palette displays a large $f(x)$ on the left-hand side.

Gamma function
A function used in advanced mathematics. For any real or complex number:

$$\Gamma(z) \equiv \int_0^\infty t^{z-1} e^{-t} dt$$

For integers, $\Gamma(z) = (n-1)!$

Graph
A frame work for displaying plots and those plots. In Theorist, a graph consists of a graph theory (a two- or three-dimensional shell) and a set of plots.

Graph Theory
A Theorist proposition. Graph theories display Line, Contour, and Surface plots, and possibly Axes and Grid lines, in a viewport.

Greek subpalette
A pop-up menu available on the variable palette for entering Greek characters. The menu displays the first three Greek characters, αβγ.

Grid lines
A Theorist plot proposition that displays a set of Grid lines in a graph theory.

HLS
A color encoding system that describes colors by their Hue, Lightness, and Saturation components

HSV
A color encoding system that describes colors by their Hue, Saturation, and Value components.

Hyperbolic functions
Functions akin to trigonometric functions, related to hyperbolas as trigonometric functions are related to circles.

Independence declaration
A Theorist proposition that declares one or more variables independent of each other or independent of all variables. By default, each variable is dependent on all other variables.

Independent variables
A variable in an equation whose value is allowed to assume values freely over a specified domain. The values of the independent variable(s) (and one or more equation) determine the values of the dependent variable(s).

Insertion point
A blinking vertical (or occasionally horizontal) bar that indicates where the next entry is placed. An insertion point does not contain a selection. You create an insertion point with the mouse, and can move it with the arrow keys and the Tab key.

Integral
A Theorist op. The integral op encloses an expression and represents a summation of infinitesimally small values.

Integration
The process of finding an integral. Integration is the inverse process of differentiation. That is, finding a function whose derivative is the original function.

Isolate
A Theorist manipulation used to rearranges an equation so that the selected expression is separated out and placed on one side of the equals sign and the rest of the equation is on the other.

Knife
A Theorist editing tool found on the right-hand side of two- and three-dimensional graph theories. Use the Knife to slice out a portion of a plot for closer inspection.

Line
A Theorist plot proposition that displays a curve (based on a selected equation) in a graph theory.

Linear
An equation or an expression that is of the first degree. The graph of a linear relationship produces a straight line (or a higher dimensional equivalent, e.g., a plane). The relationship between two expressions is linear, if the derivative of one with respect to the other is constant.

Log
A Theorist predefined name that acts as a function. It returns the common logarithm (base 10) of a given argument.

Logarithm
The power to which a base number must be raised to equal a given number. For a base of 10, log(1000) = 3; the number ten must be raised to the power of three to equal 1000.

M-Linear Operator
A Theorist class. Names of class M-Linear Operator are matrices or other ops that are associative but not commutative under multiplication.

Manipulations
Theorist commands invoked from a pull-down menu, from the keyboard, or with the mouse used to generate new, equally valid, expressions or equations from existing expressions or equations.

Matrix
A Theorist op. A matrix is a set of elements arranged in a rectangular array of rows and columns. Row vectors are horizontal matrices consisting of only one row. Column vectors are vertical matrices consisting of only one column.

Matrix subpalette
A pop-up menu available on the function palette for entering matrices of up to four-by-four elements. The menu displays the tiny grid of a three-by-four matrix.

Mod
A Theorist predefined name that behaves as a function (modulus). It returns the first number mod the second, that is, the remainder from dividing the two given arguments.

Move over
A Theorist manipulation used to move an expression from one side of an equation to the other while retaining the logical validity of the equation.

Name
A Theorist designation for a mathematical entity consisting of Roman or Greek characters, and possibly special characters.

Notebook
A Theorist file. Notebooks are displayed in windows, and contain theories, propositions, and expressions.

Number
One or more digits (and possibly an expression used for scientific notation) indicating a numeric value.

Numerical manipulations
Theorist manipulations that produce numeric (as opposed to symbolic) results. Calculate is the only numerical manipulation.

Op
A Theorist object that represents a mathematical operation (e.g., +, $\sqrt{\ }$, =). Ops join together and enclose one or more expression to create larger, more complex expressions.

Operator
The Theorist name for matrix or derivative expressions (or expressions that act like matrices or derivatives), that do not necessarily commute under multiplication.

Orientation icon
A Theorist icon displayed in the lower left-hand corner of a three-dimensional graph viewport that indicates which side of the representation of three space is displayed in the viewport.

Palette
Part of Theorist's main display. The palette is a rectangular window displayed above a notebook window and is used to enter expressions with the mouse.

Partial derivative
A Theorist op that takes derivatives with respect to a variable. It acts as a true "partial derivative" only if the specified variable is included in an appropriate independence declaration.

Pattern expression
A Theorist expression used to match the form of other expressions in order to substitute an equivalent expressions for the original.

Pi product
A Theorist op that encloses an expression and indicates the product of a series of values that the main argument takes on as a variable steps through a given domain of values. See Summation.

PICT
A Macintosh picture. The data format is based on QuickDraw graphic specifications.

Plot
A Theorist proposition that generates an image in the viewport of a two- or three-dimensional graph theory.

Polar
A coordinate system in which the position of a point is determined by the distance from a central point, and the angle of rotation from a fixed line.

Polynomial
A mathematical expression consisting of a sum of terms, each of which is the product of a constant and a variable (or variables) raised to various powers.

Power
A Theorist op (displayed as an exponent or superscript) that indicates the number of times a base expression is multiplied together.

Predefined behavior
A set of fifty-seven Theorist behaviors. Each behavior has an associated name (e.g., sin, cos), but any behavior can be assigned to any name.

Predefined name
The Theorist name associated with a predefined behavior.

Principle value
The first or primary value that results from an operation that produces more than one correct answer.

Rational function
A fraction with polynomial expressions in both the numerator and denominator.

RGB
A color encoding system that describes colors by their Red, Green, and Blue components.

Rocket
A Theorist editing tool found on the right-hand side of two- and three-dimensional graph theories. Use the Rocket to zoom in (or out) on a graph.

Root
A number (or numbers) that, when substituted into an equation for the variable(s) makes the equation a true statement. The numbers 2 and –2 are roots of the equation $x^2 = 4$.

Round
A Theorist name that acts as a function returning the integer closest to the given value.

Scalar
An expression that represents a single numeric value as distinct from a vector or matrix.

Select
A Macintosh operation of choosing, usually with the mouse, part or all of an expression or other object.

Select In
A Theorist command that selects individually all subexpressions of a selected expression.

Select Out
A Theorist command that selects the next largest enclosing structure of the current selection.

Self adjoint
An expression that is equal to the adjoint of itself.

SemiLog
A Theorist graph type that displays a logarithmic vertical axis and a linear horizontal axis.

Simplify
A Theorist manipulation that performs many actions in an attempt to reduce a selected expression to its most fundamental form.

Statement
A Theorist proposition that contains an expression or an equation. Statements that you enter (assumptions) are displayed with a square icon. Statements that the program generates (conclusions) are displayed with triangular icons.

Substitute
A Theorist manipulation that replaces a given expression with another equivalent expression.

Sum
The result of an addition.

Summation
A Theorist op that is multiplied on to an expression and indicates the sum of a series of values that the main expression takes on as a variable steps through a given domain of values.

Surface
A Theorist plot proposition that displays a surface (based on a selected equation) in a graph theory.

Symbolic manipulations
Theorist manipulations that produce symbolic (as opposed to numeric) results. All manipulation, other than Calculate, are symbolic manipulation.

Taylor series
A Theorist manipulation that produces a Taylor series expansion of a selected expression.

Theory
A Theorist work space containing propositions. Equations that are true in one theory may not be true in another. The main theory is essentially equivalent to a notebook. Case theories are subsets of a note book as are graph theories.

Transform
A Theorist manipulation that executes whatever transformation rules have been created to execute "upon transform." This manipulation is for extending and programming Theorist.

Transpose
A matrix created from a given matrix by flipping the matrix along its upper-left-to-lower-right diagonal, thereby exchanging the rows for the columns.

Trigonometric functions
Functions of angles defined as ratios of sides of right-angled triangles inscribed in a unit circle. These functions are: sine, cosine, tangent, cotangent, secant and cosecant.

Trigonometry
The branch of mathematics concerned with the ratios of triangles.

User defined behavior
A Theorist term indicating that the behavior of a name is not one of the predefined behaviors but is, instead, defined by you.

Variable
A Theorist class. Names of class Variable are scalars whose values can change dynamically without your intervention.

Variable palette
One side of the Theorist entry palette, used to enter variables and constants. This palette displays a large *x* on the left-hand side.

Viewport
Part of a graph theory. The viewport is the window on theoretically infinite two-dimensional and three-dimensional spaces. All plot propositions are displayed in the viewport.

Vector
A one dimensional matrix. All vectors are either row or column vectors, depending on their orientation.

Wildcard subpalette
A pop-up menu available on the variable palette for entering wildcard variables. The menu displays the the first three wildcard variables, @bc.

Wildcard variables
Theorist expressions that act as placeholders in an expression. Each wildcard variable matches a specific set of expressions and is used as part of pattern expressions in Substitute manipulations and transformation rules.

Working statement
A marked Theorist statement. Once a statement becomes a working statement (by your designation or the program's), it is used (if needed) by all future manipulations, and for making graphs, without notifying you. Working statements can be either assumptions or conclusions.

YIQ
A color encoding system used by broadcast television.

Bibliography

Of the many texts available about mathematics and computing, these books were used the most while developing Theorist.

Abramowitz, M., Stegun, I. A., *Handbook of Mathematical Functions*, Dover Publications, Inc., New York, 1972.

The standard book for special functions, it contains extensive formulas and tables for all of the special functions you've ever heard of (except spherical harmonics). It also has lots of useful reference material in other areas of mathematics, including combinatorial and numerical analysis formulas and tables, and Laplace transforms.

Bronshtein, I. N., Semendyayev, K. A., *Handbook of Mathematics*, 20th Edition., English translation, Hirsch, K.A., Van Nostrand Reinhold Company, New York, 1985.

This book, about the size of a small dictionary, contains almost all of the basic applied mathematics you will ever need. The type is tiny and the language is sometimes strange, having been translated from Russian, through German, into English, but the book is an amazing compendium of all of those facts that you knew at one time, but forgot, in addition to many things you never knew, at an affordable price.

Buchburger, B., Collins, G. E., Loos, R., *Computer Algebra: Symbolic and Algebraic Computation*, 2nd Edition, Springer Verlag, New York, 1982.

A comprehensive survey of the state of the art in symbolic algebra research and implementation, circa 1982. Great starting point for research.

Davenport, J. H., Siret, Y., Tournier, E., *Computer Algebra: Systems and Algorithms for Algebraic Computation*, Academic Press, 1988.

A good introduction to how computers do symbolic algebra, including some of the latest algorithms.

Gradshteyn, I. S., Ryzhik, I. M., *Table of Integrals, Series, and Products*, 4th Edition, English translation, Jeffrey, A., Academic Press, Inc., San Diego, California, 1980.

As complete a table of integrals, series, and products as you could want. Over 1000 pages.

Knuth D. E., *The Art of Computer Programming: Volume 2, Seminumerical Algorithms*, 2nd Edition., Addison Wesley Publishing Company, Reading, Mass, 1981.

Donald Knuth must be the world's authority on how to do arithmetic, in addition to many other topics. If you have forgotten how to do polynomial division modulo 13, just pick up this book for a refresher. This book contains many good algorithms for numeric, and especially symbolic, computation.

Press, W.H., Flannery, B. P., Teukolsky, S. A., Vetterling, W. T., *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press, New York, 1988.

This is a great handbook for almost all areas of numerical analysis, and a great starting point for further research. These guys have been in the trenches and know the weapons that work, those that do not, and how and when they fail. Written in a friendly style that keeps you interested.

Acknowledgements

When I wrote Expressionist, I was all by myself. So I got a lot of sympathy and help from my friends. But because Expressionist has been such a success, I've been able to build a small company that supports my user-interface dreams. The people who have joined me at Prescience have time and again worked way beyond the call of duty. Without them, Theorist would not exist. I would especially like to thank Mike Robertson, Erik Warren, Laura Wallrath, and Chester Hurtado who have worked especially hard to make this company a success.

As with Expressionist, Theorist is largely a product of the feedback I received from beta testers. It's been a long road, and only a few of you have stayed with me for the years it took to write this program, but you all contributed to the process. My sincere thanks to each of you.

I'd also like to thank Cara for putting up with me as my insanity escalated to arbitrarily high levels as the release date became arbitrarily close.

In addition, I would like to thank Mike Goddu who made it possible for this business misfit to build a million-dollar company.

Finally, I would like to thank all of the people who built the many foundations that my work rests upon, including the symbolic algebra pioneers of the 1960's and 1970's, as well as the wonderful folks who developed the tools for a truly modern user interface at Xerox PARC and Apple.

— Allan Bonadio
San Francisco
September 1990

About the Company

Prescience (pronounced PREshence) Corporation began as Allan Bonadio Associates, which was founded in July 1984 to develop graphical user-interface software for the microcomputer and workstation markets.

ABA developed MacSpell+ in conjunction with Level Four Interactive, Inc. and Creighton Development, Inc. This spelling-checking program reached number six on the Soft-Sel national best seller list and remained on the list for over six months.

The company released the Macintosh equation editor Expressionist in March 1987. Used to generate equations for technical desktop publishing, the program extends the Macintosh user interface to equations and allows WYSIWYG editing of expressions in a point-and-click manner. Expressionist has consistently been on MacAmerica's best-seller list and has become the standard equation editor for the Macintosh.

ABA incorporated as Prescience Corporation in May 1989. From its office in downtown San Francisco, the company publishes Expressionist and Theorist and contemplates new products.

President Allan Bonadio earned a B.S. in Applied and Engineering Physics from Cornell University, where he minored in computer science and psychology. A programmer for eighteen years, he has worked as a programmer, technical writer, and user-interface consultant to Apple Computer, Hewlett-Packard, NeXT, Inc., Digital Equipment Corporation, and other companies. He enjoys socializing, vacationing, cooking, psychology, scuba diving, and the history of Indo-European Languages. As the head programmer, he is currently developing new products and augmenting existing products for Prescience Corporation.

Colophon

Program design

Allan Bonadio

Programming

Allan Bonadio
Michelle Miller

Notebooks

Allan Bonadio
Alex Plotitsa

Documentation

Allan Bonadio
BC Crandall
Erik Warren

Documentation Tools

This manual was developed on several Macintosh computers, and printed on the LaserWriter II NTX. The software used to create the manual includes Theorist, Expressionist, Microsoft Word, Canvas, SuperPaint, and FlashIt.

Packaging design

Clennon Associates, Pacifica, California

Bug Report Form

Photocopy this form and use it to report bugs that you find in Theorist, or inaccuracies in the documentation. We thank you in advance for your help in making Theorist a better product.
You must be a registered user to receive technical support.

_____ _____
Name Phone or Fax number(s)

_____ _____
Address Serial number (10 digits)

_____ _____
City State Zip or Postal Code Country

The problem is with: (check all that apply)
- ❑ The Theorist application; version # (check About box): _____
- ❑ The documentation; manual title and page: _____
- ❑ Printed output; printer make and model: _____
- ❑ Using Theorist with other programs (describe on reverse)

I discovered the problem while using: (check all that apply)
- ❑ Mac Plus ❑ Mac SE ❑ Mac SE/30 ❑ Mac Portable
- ❑ Mac II ❑ Mac IIx ❑ Mac IIcx ❑ Mac IIci
- ❑ Other Macintosh model: _____

(Theorist does not run on the Mac 128K, 512K, 512KE, XL, or the Lisa.)

Please describe the bug in as much detail as possible (on a separate sheet of paper, or the back of this page). What goes wrong? What were you trying to do when the bug appeared? Are you using MultiFinder? How much RAM does your machine have? (Theorist requires at least one megabyte.) What is your hardware set up (RAM disks, accelerator boards)? Did the bug appear just once, or repeatedly?

We are also open to general (and specific) suggestions. Write us a letter; tell us what you'd like to see in the program. (No guarantees.)

Send to: FAX: (415) 882-0530
Prescience Corporation, 939 Howard Street, San Francisco, CA 94103

Theorist® License and Warranty

In consideration of payment of the License fee, which is a part of the price you paid for this product, Prescience Corporation (Prescience), as Licensor, grants to you, the Licensee, a nonexclusive right to use and display this copy of a Prescience software program on a single computer at a single location. Prescience reserves all rights not expressly granted to Licensee. As Licensee, you own the magnetic media on which the software is originally or subsequently recorded. Prescience retains title and ownership of the software recorded on the original disk copies and all subsequent copies of the software, regardless of the form or media in or on which the original and other copies may exist. This license is not a sale of the original software or any copy.

YOUR DISK IS CONSIDERED PROOF OF PURCHASE AND YOU MUST KEEP IT FOR UPGRADES AND UPDATES.

This software and the accompanying written materials are copyrighted. Unauthorized copying of the software, including software that has been modified, merged, or included with other software, or of the written materials is expressly forbidden. YOU MAY BE HELD LEGALLY RESPONSIBLE FOR ANY COPYRIGHT INFRINGEMENT THAT IS CAUSED OR ENCOURAGED BY YOUR FAILURE TO ABIDE BY THE TERMS OF THIS LICENSE. Subject to these restrictions, you may make one (1) copy of the software solely for backup purposes. You must reproduce and include the copyright notice as part of the backup copy.

As the licensee, you may transfer the software from one computer to another provided that THE SOFTWARE IS USED ON ONLY ONE COMPUTER AT A TIME. You may not electronically transfer the software over a network or distribute copies of the software or accompanying written materials to others. You may not modify, adapt, translate, reverse engineer, decompile, disassemble, or create derivative works based on the software or the written materials, except that excerpts of the written materials may be quoted in software reviews.

This software is licensed to only you, the Licensee, and may not be transferred to anyone without the written permission of Prescience. This license will terminate automatically without notice from Prescience if you fail to comply with any provision of this license. This license is effective until terminated. Upon termination, you must destroy all copies of software and written materials.

Prescience warrants that the Theorist computer software recorded on the disks will perform substantially in accordance with the specifications set forth in the documentation provided with the software. Prescience does not warrant that the functions contained in the software will meet your requirements or that the operation of the software will be uninterrupted or error free.

Prescience warrants the Theorist disks to be free of defects for 90 days from the purchase date. If the disk is found to be defective in this 90 day period, return the disk to Prescience for a free replacement. This warranty does not cover software or media that has been altered in any way.

ANY WARRANTIES IN THIS AGREEMENT ARE LIMITED TO 90 DAYS FROM DATE OF PURCHASE. PRESCIENCE IN ANY CASE SHALL NOT BE LIABLE FOR ANY DAMAGES, INCLUDING ANY LOST PROFITS, LOST SAVINGS, OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING FROM BREACH OF WARRANTY, NEGLIGENCE, OR OTHER CAUSE. OUR LIABILITY IS LIMITED TO THE EXPRESS WARRANTIES DESCRIBED IN THIS DOCUMENT. IN NO CASE SHALL LIABILITY EXCEED THE PURCHASE PRICE OF THE PRODUCT. Some states do not allow such limits on liability so the above limits may not apply to you.

This agreement is governed by the laws of the state of California.

Index

! (factorial op) 75
< 72
> 72
≤ 72
≥ 72
≠ 72
′ 59
⟨ 59
⟩ 59
∇ 59
ℜ 59
ℵ 59
ℑ 59
π 64
∞ 57, 59, 64

A

Abbreviating expressions
  collapsars 83
Absolute value op 74
Absolute values
  entering 74
Accuracy (*see* resolution)
Add Axes command 141, 162
Add Contour Plot command 141, 160, 162, 220
Add Grid Lines command 141
Add Line Plot command 141, 162, 215
Add Surface Plot command 124, 141, 162
Adding
  axes 147
Addition op 71
Adjoint op 75
Adjoints
  entering 75
Adjusting
  graph bounds 266
  graphs with mouse 141, 162
  independence declarations 43
  shade value 184
  shade value range 185, 186
Algebra
  non-commutative 54
Algorithms
  discussing 5
  numeric precision 5
Animate command 194
Animation 193-197
  dialog box 194
  example files 302, 304
  memory for 195, 197
  number of frames 195
  playback speed 195
  Projectionist 299
  rotating 266
Animation variable 193
Animations
  saving 195
Apply 116-119
  example of 116, 117, 119
  using 116, 117, 119
  with equations 116, 117
  with fractions 118
  with products 119
  with single terms 117
Apply command 116
Arbitrary constants
  changing defaults 91
  examples of 91
Arbitrary Constants option 91
Arccos (ArcCosine)
  predefined function 66
Arccosh (Hyperbolic arccosine)
  predefined function 66
Arccot (ArcCotangent)
  predefined function 66

Arccoth (Hyperbolic cotangent)
predefined function 66
Arccsc (ArcCosecant)
predefined function 66
Arccsch (Hyperbolic arccosecant)
predefined function 66
Arcsec (ArcSecant)
predefined function 66
Arcsech (Hyperbolic arcsecant)
predefined function 66
Arcsin (ArcSine)
predefined function 66
Arcsinh (Hyperbolic arcsine)
predefined function 66
Arctan (ArcTangent)
predefined function 66
Arctanh (Hyperbolic arctangent)
predefined function 66
Arguments
function 56, 73
Arizona
Custom color group 187
Aspect ratio (3D) 167
Assumption propositions 26, 28
Assumptions
creating 26, 28
figure of 26
Audience v
Auto Casing 7
and Isolate 95
and maximum statements 30
and minimum statements 30
case theories 32
effects of 91
turning on and off 91
Auto Casing command 32
Auto Casing option 91
Auto Collapsar option 83
Auto Simplify
and Collect 113
and Simplify 108
effects of 90
possible problems with 108
turning on and off 91
Auto Simplify option 91
Auto-Redraw
graph preference 201
Axes 147
adding 147
figure of (2D) 144, 146
figure of (3D) 166, 169
moving 146, 169
plot propositions 146, 169
Axes (2D) 146
Axes (3D) 169
Axis labels
graph preference 202

B

Base 10 logarithm 65
Base *e* logarithm 65
Behavior
types of 60
Bibliography 325
Bitmapped graphs
preference 200
bottom
predefined constant 68
Brackets 84
Bug report form 330
Bundles
copying 21
notebooks 21

C

Calculate 101-104
conditional ops 80
definite integrals 102
displayed precision 102
evaluate at ops 80
examples of 102, 103
integrals 78
paramatrized path 102
partial derivatives 77, 102

Calculate (continued)
pi products 79
precision of 102
Uncalculate 104
using 101
Calculate command 104
Calculus
functions 65
Case sensitivity 60
Case statements
conditional op 79
Case theories 32–35
and finding roots 159
creating 33
figure of 32
generating 32
in graph theories 35
searching 34
vs arbitrary constants 32
vs graph theories 35
Ceiling
predefined function 64
Changing
expressions 223
graph bounds 144, 164
name behavior 41
name class 41
name declarations 60
name spelling 41
working statements 28
Characters
for names 59
Greek 59
Roman 59
Clarify command 40, 52
Class
behavior 41
changing 41
Constant 53
D-Linear Operators 54
declaring 52
declaring a name's 59
default 52
determining 52
erroneous 52
Function 55
M-Linear Operator 54
multiplying different 71
of existing expression 52
of expressions 52
of predefined names 53
types of 52
Variable 53
Class system 52
Close command 19
CMY color model 190
Co-Processors
floating point 4
Math 4
Collapsars 83
and addition 83
and multiplication 83
Collapse All command 48, 83
Collapse command 48, 83
Collapsing
propositions 48
Collect 112-114
and Auto Simplify 113
and transformation rules 45, 121
examples of 112
multivariate polynomials 114
using 112
Collect command 114
Color 3D graphs
creating 174
description of 139
figure of 174
plot proposition for 175
Color groups
for Custom color scheme 187
Color models
Cyan, Magenta, Yellow 190
Hue, Lightness, Saturation 191
Hue, Saturation , Value 191
Red, Green, Blue 190
table of direct 192
YIQ 191

Color schemes 183-192
Complex 175, 188
Custom 186
Direct 190
Gradient 184
list of 183
Solid 176, 183
Surface plots 172
Color spectrum
using 184
Colors
changing graph 175, 176
Macintosh color wheel 176
Column Number command 268
ColumnsOf (matrix)
predefined function 67
Commands
Add Axes 141
Add Contour Plot 141, 160, 162, 220
Add Grid Lines 141
Add Line Plot 141, 162, 215
Add Surface Plot 124, 141, 162
Animate 194
Apply 116
Auto Casing 32
Calculate 104
Clarify 40, 52
Close 19
Collapse 48, 83
Collapse All 48, 83
Collect 114
Column Number 268
Commute 97
Copy 58, 237
Copy as PICT 38, 58, 202, 229
Copy Heights Array 204
Copy Points List 205
Delete 234
Display Precision 57
Expand 112
Expose 48, 83
Expose All 48, 83
Factor 115
Find Root 159
Generate EPS File 203
Generate PICT File 202
Get Info 11, 25, 26, 52, 60
Independence Decl. 44
Integrate by Parts 125
Isolate 94
Make Working Stmt 28, 55
Move Over 96
New Notebook 19
Open 19
Page Setup 11
Paste 237
Print 10
Revert 19
Row Number 268
Save 19
Save as 19
Save as Stationary 19
Select In 225, 233, 239
Select Out 225, 233
Simplify 108
Spherical 3D 219
Stop Working Stmt 28
Substitute 99
Taylor Series 123
Transform Rule 46, 122
Comment propositions 37
creating 225
Comments 37
changing fonts 37
changing size 37
editing 241
formatting 37
picture 38
text 37
Common logarithm 65
Commute 96
by hand 97
examples of 98
figure of 97
partial derivatives 98
using 96
Commute command 97

Commuting
D-Linear Operators 55
Complex
variable in graphs 173
Complex 3D graphs
creating 173
description of 138
figure of 173
plot proposition for 173
Complex color scheme 188
possible settings 188
Complex colors
using 188
Complex numbers
as exponents 73
as subscripts 74
using 57
Computer systems 4
Computers
reliability 7
Conclusion propositions 26, 28
Conclusions
figure of 26
replacing assumptions with 92
Conditional ops 79
Calculate 80
entering 80
Simplify 80
Connections
between propositions 25
Constants
class 53
creating 53
description of 139
examples of 53
from variables 262
using 53
Contour graphs
creating 157
figure of 157
plot proposition for 157
Contour plots
altitude variable 150
density settings 150
description of 150
example of 151
figure of 150
location vector 150
making spherical 219
Contour style 150
Copy as PICT command 38, 58, 202, 229
Copy command 58, 237
Copy Heights Array command 204
Copy Points List command 205
Copying
bundles 21
expression as bit map 58
expressions 58
expressions as text 58
name conflicts 21
numeric values 58
Cos (Cosine)
predefined function 66
Cosh (Hyperbolic cosine)
predefined function 66
Cot (Cotangent)
predefined function 66
Coth (Hyperbolic cotangent)
predefined function 66
CPUs
acceptable 4
Creating
assumptions 26
case theories 33
Color 3D graphs 174
Complex 3D graphs 173
constants 53
comment propositions 225
Contour graphs 157
Cylindrical 3D graphs 180
D-Linear Operators 54
del 258
Density graphs 156
derivatives 77
differentials 255
Direct color scheme 190

Creating (continued)
evaluate at op 258
expressions 242
Functions 55, 252
graph animations 193-197
graph theories 136
graphs 136
graphs from scratch 215
Illuminated 3D graphs 176
Independence declarations 44
integrals 255
iterations 257
Linear graphs 153
Linear operators 253
M-Linear Operators 54
matrices 254
name declarations 59
names 245
numbers 244
ops 246
Parametric graphs 152
Polar graphs 155
propositions 26
ranges 258
relational ops 258
SemiLog graphs 154
Spherical 3D graphs 178
stationary files 19
Taylor Series 123
three-dimensional graphs 171-180
transformation rules 46
two-dimensional graphs 148-158
Variables 53
working statements 28
Zero Contour graphs 158
Cropping the viewport (2D) 145
Cropping the viewport (3D) 167
Cross product op 77
Cross products
entering 77
Csc (Cosecant)
predefined function 66
Csch (Hyperbolic cosecant)
predefined function 66
Curly braces 84
Curves
integrating spiral 102
Custom color
list of color groups 187
Custom color group
Arizona 187
Kansas 187
Louisiana 187
New York 187
Oregon 187
Virginia 187
Wyoming 187
Custom color scheme 186
possible settings 175, 186
Custom colors
using 186
Customizing Theorist 19, 20
Cyan, Magenta, Yellow
color model 190
Cylindrical
functions 67
Cylindrical 3D graphs
creating 180
description of 140
figure of 180
plot proposition for 180

D

d (Differential operator)
predefined operator 65
D-Linear Operators
and independence declarations 42
class 54
commuting 55
creating 54
determining class of 55
examples of 55
multiplication 72
using 54

Data
exporting 9, 203
importing 9
Data-point values
exporting 9, 203
importing 9
Decimal numbers 57
Declarations
figure of (2D) 145
figure of (3D) 168
independence 42
Declarations proposition 19
Declaring
behavior of a name 59
class 52
class of a name 41, 59
name class 41
names 40, 59
spelling of a name 59
Default
class 52
name declarations 40
predefined names 62
Definite integrals
Calculate 102
Definition of
graph bounds (2D) 144
graph bounds (3D) 166
viewport controls (2D) 145
viewport controls (3D) 167
Del
creating 258
Deleting
example of 234
figure of 234
matrix rows 239
protected propositions 21
Deleting propositions 21
Density graphs
creating 156
description of 139
figure of 156
plot proposition for 156
Dependence of variables 42
Dependent variables
in graphs 137
Derivatives
creating 77
increment value 30
independence declarations 42
Details
graph 134
Details box
in 2D graphs 143
in 3D graphs 165
Details button
figure of 132, 135
Details icon 136
Determining
class 52
class of D-Linear Operators 55
class of matrices 54
Dialog box
animation 194
Graph preferences 199
name declarations 41
Taylor Series 123
Transform Rule 121
variable dependence 137
Differential operator (d) 65
Differentials
creating 255, 256
entering 255, 256
Direct color models
list of 190
Direct color scheme 190
creating 190
possible settings 190
Direct colors
Cyan, Magenta, Yellow 190
Hue, Lightness, Saturation 191
Hue, Saturation , Value 191
Red, Green, Blue 190
table of 192
using 190
YIQ 191
Display Precision option 57, 102

Display technology
setting 199
Distribution files
animation 302, 304
list of 306
Distribution notebooks
Hyperbolic Functions 22
Log and Power Functions 22
Standard Rules and Declarations 22
Trig Functions 22
using 21

Dithering
graph preference 201
Division
by zero 72
Division op 72
Domains
of graph variables 213
Dot product op 76
Dot products
entering 76
Duplicate name declarations 21

E

e 64
east
predefined constant 68
Edit menu 274
Close 19
Copy 58, 237
Copy as PICT 38, 58, 202, 229
New Notebook 19
Open 19
Paste 237
Save 19
Save as 19
Save as Stationary 19
Select In 225, 233, 239
Select Out 225, 233
Editing 223-258
basics 225
Comments 241
deleting 234
expressions 237, 250
graph bounds 266
graph bounds (2D) 144
graph bounds (3D) 166
graphs 141, 162
matrices 239, 267
N-ary expressions 239
names 60, 239
on the left 267
ops 225
propositions 237
Select In 225
Select Out 225
Editing buttons
figure of 132, 135
Editing icons
details box 143, 165
resolution 142, 164
Knife 142, 164
list of 136
orientation icon 165
Rocket 142, 164
size box 143, 165
ElementsOf (matrix)
predefined function 67
Embedded PostScript
graph preference 201
Entering
absolute values 74
adjoints 75
conditional ops 80
cross product 77
del 258
differentials 255, 256
dot product 76
evaluate at ops 80, 258
exponents 74
expressions 237
factorials 75
functions 73, 252
Greek characters 231

Entering (continued)
integrals 255
integrals with limits 78
integrals without limits 78
iterations 257
Linear operators 253
matrices 76, 232, 254
minus sign 251
names 245
numbers 244
ops 246
partial derivatives 77
pi products 79
powers 74
range ops 80
ranges 258
relational ops 73, 258
square roots 75
subscripts 74
summations 79
vectors 76
wildcard variables 81, 232
Equals op 72
Errors
round off 6
Escape key 230
Escape levels 230
Evaluate at op 80
Calculate 80
creating 258
entering 80, 258
Simplify 80
Evaluating expressions
Calculate 101
Example notebooks 21
Example of
Apply 116, 117, 119
Calculate 102, 103
Collect 112
Class Constant 53
Class D-Linear Operators 55
Class Function 56
Class M-Linear Operators 54
Class Variable 53
Commute 98
Contour plot 151
Expand 109, 110
Factor 115
Fortranish 229
Independence declarations 43
Integrate by Parts 125, 127
Isolate 94, 95
Move Over 96
Multidimensional Taylor Series 123
Simplify 106, 107
Projectionist 302, 304
scientific notation 57
Transform Rule 120
two Surface plots 125
typing into selections 242
wrong class 52
Uncalculate 105
Exp (exponential)
predefined function 65
Expand 109-112
examples of 109, 110
integrals 78
pi products 79
summations 78
transformation rules 45, 121
using 109
vs MiniExpand 109
Expand command 112
Expanding collapsed expressions 83
Exponential function (exp) 65
Exponents
complex numbers 73
entering 74
irrational numbers 73
power op 73
syntax 73

Exporting
data 9
graph data 203
graphics 9
graphs 202
graphs as EPSF 203
graphs as PICT images 202
graphs as PICT files 202
text 9
Expose All command 48, 83
Expose command 48, 83
Exposing
propositions 48
Expressionist
using with Theorist 305
Expressions 51
changing 242
class of 52
Constants 53
copied in Fortranish 229
creating 242
D-Linear Operators 54
editing 223, 237, 250, 267
enclosing in parentheses 263
entering 237
escape levels 230
functions 55
grouping 84
M-Linear Operators 54
names 59
numbers 57
ops 69
precedence levels 226
selecting 236, 238
structure of 224
types of 51
Variables 53

F

Factor 114-115
examples of 115
Factor command 115
Factorial op 74
Factorials
entering 75
Factoring
multivariate polynomials 264
Figure of
3D graph 216, 218
Axes (2D) 144, 146
Axes (3D) 166, 169
case theories 32
Color 3D graph 174
Commute manipulation 97
Complex 3D graph 173
Contour graph 157
Contour plot 150
Cylindrical 3D graph 180
Declarations comment (2D) 145
Declarations comment (3D) 168
deleting an expression 234
Density graph 156
details button 132, 135
editing buttons 132, 135
functions 31
graph bounds (2D) 144
graph bounds (3D) 166
graph details (2D) 144
graph details (3D) 166
graph theory icon (2D) 132
graph theory icon (3D) 135
Grid lines (2D) 144, 147
Grid lines (3D) 166, 170
Illuminated 3D graph 176
Independence declarations 42
Isolate manipulation 94
Line plot 149
Linear graph 153
name declaration 39, 59
notebook 18
orientation icon 135
palette 231, 232
Parametric graph 152
picture comments 38

Figure of (continued)
plot propositions (2D) 144
plot propositions (3D) 166
Polar graph 155
SemiLog graph 154
size box 135
Spherical 3D graph 178
statements 26
Substitute 98, 99
Surface plot proposition 172, 183, 184, 186, 188, 190
three-dimensional graph 135
transformation rule 45
two-dimensional graph 132
two-dimensional graph details 134
viewport 132, 135
viewport controls (2D) 144
viewport controls (3D) 166
working statements 28
Zero Contour graph 158
Figures
pasting in 38
File menu 272
Get Info 11
Page Setup 10
Revert 19
Files
notebooks 18
stationary 19
Find Roots command 159
Finding
name declarations 60
Finding roots 159-161
Floating point
calculations 6
co-processors 4
round-off errors 6
Floor
predefined function 64
Formatting
comments 37
propositions 47
FORTRAN syntax
used to copy expressions 229
Fortranish 226
examples of 229
Frames
animation 195
Freely rotating
3D graphs 163
FromCylindrical
predefined function 67
FromPolar
predefined function 67
FromSpherical
predefined function 67
Function call op 73
Function palette 232
Function statement 31
Functions
arguments 31, 73
calculus 65
class 55
creating 55, 252
cylindrical 67
entering 73, 252
examples of 56
figure of 31
hyperbolic 66
matrix 67
multiple arguments 56
numeric 64
polar 67
powers of 73
spherical functions 67
trigonometry 66
using 55, 73
vector arguments 56
vs statements 73
wildcard variables 31

G

Gamma (Γ)
  predefined function 65
  numeric precision 5
Garbage in
  garbage out 12
Generate EPS File command 203
Generate PICT File command 202
Get Info command 11, 25, 26, 52, 60
GIGO 13
Glossary 309
Gradient color scheme 184
  possible settings 184
Gradient colors
  using 184
Graph bounds 213
  3D graphs 164
  and rotating 166
  balancing 266
  bottom 68
  changing 144
  changing with Knife 142, 164
  changing with Rocket 142, 164
  definition of (2D) 144
  definition of (3D) 166
  east 68
  figure of (2D) 144
  figure of (3D) 166
  left 68
  north 68
  predefined names 68
  radius 68
  right 68
  setting maximum 30
  setting minimum 30
  south 68
  top 68
  west 68
Graph details
  comparison of 211
  declarations comment (2D) 145
  declarations comment (3D) 168
  description of 133
  figure of (2D) 144
  figure of (3D) 166
  plot propositions in (2D) 145
  plot propositions in (3D) 168
  three-dimensional 166-171
  two-dimensional 144, 148
Graph menu 282
Graph option 199
Graph preferences 199
  Auto-Redraw 201
  axis labels 202
  bitmapped vs objects 200
  dialog box 199
  display technology 199
  dithering 201
  Embedded PostScript 201
Graph programming 209-220
  introduction to 210
Graph theories 36, 132-140
  as shells 141, 162
  creating 136
  description of 133
  vs case theories 35
  with other theories 140
Graph theory icon (2D)
  figure of 132
Graph theory icon (3D)
  figure of 135
Graphics
  importing 9
Graphs 131-220
  adjusting 141, 162
  animating 193-197
  changing color of 175, 176
  changing view of 141, 162
  Color 3D 139, 174
  color schemes 183-192
  comparing details of 211
  Complex 3D 138, 173

Graphs (continued)
Contour 139, 157
creating 136
custom 210
Cylindrical 3D 140, 180
Density 139, 156
dependent variables 137
designing 210
details, figure of 134
"disk" plot 217
editing 141, 162
exporting 9, 202
exporting as EPSF 203
exporting as PICT 202
exporting as PICT files 202
exporting data of 203
figure of 216, 218
from scratch 215
graph bounds 213, 266
Illuminated 3D 139, 176
independent variables 137
Line plot in three space 215
Linear 138, 153
list of 2D 148
list of 3D 171
location vector 213
memory needs 200
Parametric 152
Polar 138, 155
printing 198
programming 209-220
Projectionist 299
rendering speed 4
reordering dependencies 217
resolution of 142, 164
rotational freedom 163
SemiLog 138, 154
Spherical 3D 140, 178
spherical contour plots 219
subscripted variables 266
Three dimensional 162-192
Two dimensional 141-158
viewing area (2D) 144
viewing area (3D) 166
Zero Contour 139, 158
Graphs menu
Add Axes 141
Add Contour Plot 141, 160, 162, 220
Add Grid Lines 141
Add Line Plot 141, 162, 215
Add Surface Plot 124, 141, 162
Animate 194
Copy Heights Array 204
Copy Points List 205
Find Root 159
Generate EPS File 203
Generate PICT File 202
Spherical 3D 219
Greek characters
entering 231
Grid lines
appearance of 148, 171
figure of (2D) 144, 147
figure of (3D) 166, 170
plot propositions (2D) 147
plot propositions (3D) 170
spacing of (2D) 147
spacing of (3D) 170
Grouping expressions 84

H

Hardware
required 4
Hiding propositions 48
Higher resolution icon 136
HLS color model 191
HSV color model 191
Hue, Lightness, Saturation color model 191
Hue, Saturation , Value color model 191
Hyperbolic functions 66
Hyperbolic Functions distribution notebook 22

I

i 64
Icons
  Details 136
  Higher resolution 136
  Knife 136
  Lower resolution 136
  open-hand 141, 162
  Rocket 136
Identity matrices 268
Illuminated
  optical characteristics 182
Illuminated 3D graphs
  creating 176
  description of 139
  figure of 176
  plot proposition for 176
Im
  predefined function 64
Imaginary numbers
  predefined name (*i*) 64
Importing
  data 9
  data-point values 9, 241
  graphics 9, 38
  text 9
Increment statement 30
Increment value
  definite integrals 102
  partial derivatives 77, 102
Independence Decl. command 44
Independence declarations 42
  adjusting 43
  and derivatives 42
  and M-Linear Operators 54
  and multiple integrations 43
  and Taylor expansions 43
  creating 44
  example 43
  figure of 42
  for D-Linear Operators 42
  for M-Linear Operators 42
  for Taylor Series 124
  iteration variables 43
  precedence 44
  using 43
  variables in multiple 44
Independent variables
  in graphs 137
Index (subscript) op 74
Infinity 64
  entering 57, 245
Infix ops 228
Input menu 276
  Column Number 268
  Independence Decl. 44
  Row Number 268
  Transform Rule 46
Integers
  arbitrary constants 91
Integral op 78
Integrals
  calculating 78
  creating 255
  entering 78, 255
  Expand 78
  Simplify 78
Integrate by Parts 125-127
  example of 125, 127
  using 125
Integrate by Parts command 125
Integrating
  and Auto Simplify 108
  spiral curve 102
Integration
  by parts 125
  numeric 78
  numeric precision 5
Interface 5
Invoking
  transformation rules 45, 121
Irrational numbers
  as exponents 73
  as subscripts 74

Isolate 93-96
  and arbitrary constants 92
  and Auto Casing 95
  examples of 94, 95
  figure of 94
  using 93
Isolate command 94
Iteration variables
  independence declarations 43
Iterations
  creating 257
  entering 257

K

Kansas
  Custom color group 187
Key words
  list of 309
Keyboard maps 289
Knife
  changing graph bounds 142, 164
  in 2D graphs 142
  in 3D graphs 164
Knife icon 136

L

Largest number 57
Launching Theorist 19
left
  predefined constant 68
Lens setting
  viewport 167
Letters
  for names 59
Levels of escape 230
License 331
Line plots
  description of 149
  figure of 149
  location vector 149
  types of graphs in 148
Line style 149
Linear graphs
  creating 153
  description of 138
  figure of 153
  graph details 211
  plot proposition for 153
Linear operators
  creating 253
  entering 253
List of
  2D graphs 148
  3D graphs 171
  color schemes 183
  command manipulations 101
  Complex shade values 189
  Custom shade values 188
  Direct color models 190
  distribution files 306
  editing icons 136
  escape levels 230
  Gradient shade values 186
  key words 309
  manipulations 89
  optical characteristics 182
  Surface plots 181
Ln (Natural log)
  predefined function 65
Location vector
  Contour plots 150
  for polar plots 213
  Line plots 149
  plot propositions 147
  plot propositions (2D) 146
  plot propositions (3D) 169
  Surface plots 172
Log
  predefined function 65
  subscripts 61
Log and Power Functions
  distribution notebook 22
Logarithm function (log) 65
Logical dependence 27

Long expressions
collapsing 83
Louisiana
Custom color group 187
Lower resolution icon 136
M-Linear Operators
and independence
declarations 42
class 54
creating 54
examples of 54
multiplication 72
using 54

Mac II version 4
differences 4
floating point precision 4
Macintosh color wheel 176
Main theory proposition 25
Make Working Stmt
command 28, 55
Managing notebooks 20
Manipulate menu 280
Apply 116
Calculate 104
Collect 114
Commute 97
Expand 112
Factor 115
Integrate by Parts 125
Isolate 94
Move over 96
Simplify 108
Substitute 99
Taylor Series 123
Transform Rule 122
Manipulating
assumptions 26
conclusions 26
expressions 26
in place 92
Manipulations 89-127
and maximum statements 30
and minimum statements 30
Apply 116-119
Calculate 101-104
Collect 112-114
commands 101-127
list of 101
Commute 93, 96
Expand 109-112
Factor 114-115
hand 93-100
Integrate by Parts 125-127
Isolate 93, 96
list of 89
Move Over 93, 96, 262
options 90
other 120
preference
Auto Simplify 90
preferences 90
Simplify 105-108
Substitute 93, 98-100
Taylor Series 122
Transform Rule 120-122
trustworthy 7
Uncalculate 104
valid 7, 26
vs ops 69
Maps
keyboard 289
Matching
wildcard variables 82
Math co-processors 4
Mathematical perfection 7
Matrices
creating 254
creating identity 268
determining class of 54
editing 239, 267
entering 76, 232, 254
functions 67
importing data 241
indexed 74
multiplication of 75

Matrices (continued)
numeric 57
selecting 239
subscripts for 60
subscripts of 74
Matrix op 75
Maximum statement 30
Memory
for animations 195, 197
for graphs 200
running out of 4
Menus 271-286
Mesh/No Mesh
Surface plots 172
MiniExpand 109
Minimum hardware 4
Minimum statement 30
effect on manipulations 30
Minus sign
using 251
Mod
predefined function 64
Modifying
expressions 242
More info
for propositions 60
More Info button 26
Move Over 96
example of 160
examples of 96
using 96
Move Over manipulation 96, 262
Moving
Axes 146, 169
propositions 47
Multidimensional Taylor Series
example of 123
Multiple integrations
and independence declarations 43
Multiple selections 238
Multiple solutions 7
Multiplication
D-Linear Operators 72
M-Linear Operators 72
operand class 71
Multiplication op 71
N-ary expressions
selecting 238
Name behavior
changing 41
Name class
changing 41
Name declaration
figure of 39
Name declarations 39
changing 60
creating 59
default 40
dialog box 41
duplicates 21
figure of 59
finding 60
location of 40
placement of 59
Names 59
and wildcard variables 41
case sensitivity 60
changing behavior of 63
changing spelling of 41, 60
characters for 39, 59
creating 245
declaring 40
declaring class of 41
entering 245
for predefined behaviors 59
letters for 59
predefined 62
selecting 239
subscripted 60
symbols for 59
types of 39
user defined 41
using predefined behaviors 41
NaN values 57
Natural log
base of (*e*) 64
subscripts 61

Natural logarithm (ln) 65
Negation op 71
tip for 262, 267
using 251
with numbers 57
Negative numbers 57
New Notebook command 19
New York
Custom color group 187
Non-commutative algebra
example 54
north
predefined constant 68
Notebook menu 278
Clarify 40, 52
Collapse 48, 83
Collapse All 48, 83
Expose 48, 83
Expose All 48, 83
Get Info 25, 26, 52, 60
Make Working Stmt 28, 55
Stop Working Stmt 28
Notebooks 17-22
bundles 21
changing default 20
collapsing propositions 48
default display 18
distribution 21
example 21
figure of 18
files 18
managing 20
multipage 10
new 18
outlining 47
printing 10
propositions 25
special 19
Standard Rules and Declarations 22
stationary 20
Numbers 57
complex 57
copying out 58
creating 244
decimal 57
display precision 57
entering 244
infinity 57
largest 57
negative 57
overflow 57
scientific notation 57
smallest 57
storage precision 57
undefined 57
underflow 57
Numeric
functions 64
integration 78
matrices 57
Numeric values
precision of 57
Numerical
pi products 79
subscripts 60
summations 78

O

Object-oriented graphs
preference 200
Opaque
optical characteristics 182
Open command 19
Open-hand icon 141, 162
Operator hierarchy
table of 227
Ops 69-80
absolute value 74
addition 71
adjoint 75
binary 69
conditional 79
creating 246
cross product 77
division 72
dot product 76

Ops (continued)
editing 225
entering 246
equals 72
escape levels
list of 230
evaluate at 80
factorial 74
function call 73
hierarchy of 227
in statements 26
index (subscript) 74
infix 228
integral 78
matrix (vector) 75
multiplication 71
n-ary 69
negation 71
partial derivative 77
pi product 79
postfix 228
power (exponent) 73
prefix 227
range 80
relational 72
square root 75
subtraction 71
summation 78
symbols for 69
table of 69
unary 69
vs manipulations 69
Optical characteristics
description of 182
Illuminated 182
list of 182
Opaque 182
Surface plots 172
Translucent 182
Transparent 182
Options
Arbitrary Constants 91
Auto Casing 91
Auto Collapsar 83
Auto Simplify 91
Display Precision 57, 102
Graph 199
Show Palette 233
Order statement 31
Oregon
Custom color group 187
Organizing
propositions 47
Orientation icon
changing graph bounds 165
figure of 135
in 3D graphs 165
Orthogonal dimensions
and independence
declarations 43
Other manipulations 120
Outlining
propositions 47
Output
notebooks 10
Overflow
value for 57

P

Page Setup command 11
Palette 231
figure of 231, 232
function 232
variable 231
Paramatrized path
integrating 102
Parametric graphs
creating 152
figure of 152
plot proposition for 153
Parentheses 84
enclosing expressions 263
Partial derivative op 77

Partial derivatives
and Commute 98
Calculate 77, 102
calculating 77
entering 77
increment value 77
Simplify 77
Paste command 237
Pattern
Substitute 98
transformation rules 45
Pattern matching
wildcard variables 82
Pi 64
Pi Product op 79
and maximum statements 30
and minimum statements 30
Pi products
Calculate 79
entering 79
Expand 79
numerical 79
PICS files 195
PICT 1 199
PICT 2 199
Picture
copying expression as 58
Picture comment propositions 37
Picture comments 38
figure of 38
Plot propositions 136
Axes (2D) 146
Axes (3D) 169
Contour plots 150
figure of (2D) 144
figure of (3D) 166
for Color 3D graphs 175
for Complex 3D graphs 173
for Contour graphs 157
for Cylindrical 3D graphs 180
for Density graphs 156
for Illuminated 3D graphs 176
for Linear graphs 153
for Parametric graphs 153
for Polar graphs 155
for SemiLog graphs 154
for Spherical 3D graphs 178
for Zero Contour graphs 158
Grid lines (2D) 147
Grid lines (3D) 170
introduction to (2D) 146
introduction to (3D) 169
Line plots 149
Surface 72, 181-192
with Complex coloring 188
with Custom coloring 186
with Direct coloring 190
with Gradient coloring 184
with Solid coloring 183
Plot resolution
definition of 142, 164
Plots
how generated 147
how generated (2D) 146
how generated (3D) 169
resolution of 142, 164
Polar
functions 67
Polar graphs
creating 155
description of 138
figure of 155
graph details 211
plot proposition for 155
radius 214
Polynomials
and Collect 114
factoring multivariate 264
solving quadratic 94
postfix ops 228
PostScript
and animation files 196
embedded in graphs 201
EPS files 203
Translucent plots 182
Transparent plots 182
Power (exponent) op 73

Precedence hierarchy 226
Precision
  digits of 6
  display 57
  Mac II version 4
  numeric 5
  storage 57
Predefined
  names 62
Predefined behaviors
  and predefined names 62
  assigning to new names 62
  table of 64
  using other names 41
Predefined constants
  e 64
  i 64
  infinity 64
Predefined functions
  arccos 66
  arccosh 66
  arccot 66
  arccoth 66
  arccsc 66
  arccsch 66
  arcsec 66
  arcsech 66
  arcsin 66
  arcsinh 66
  arctan 66
  arctanh 66
  ceiling 64
  ColumnsOf (matrix) 67
  cos 66
  cosh 66
  cot 66
  coth 66
  csc 66
  csch 66
  ElementsOf (matrix) 67
  exp 65
  floor 64
  FromCylindrical 67
  FromPolar 67
  FromSpherical 67
  gamma (Γ) 65
  Im 64
  ln 65
  log 65
  mod 64
  Re 64
  round 64
  RowsOf matrix 67
  sec 66
  sech 66
  sin 66
  sinh 66
  tan 66
  tanh 66
  ToCylindrical 67
  ToPolar 67
  ToSpherical 67
Predefined names
  and predefined behaviors 62
  associated behaviors 62, 64
  bottom 68
  calculus functions 65
  changing spelling of 63
  class of 53
  cylindrical functions 67
  default set 62
  east 68
  graph bounds 68
  hyperbolic functions 66
  left 68
  matrix functions 67
  north 68
  numeric functions 64
  polar functions 67
  radius 68
  right 68
  south 68
  spherical functions 67
  table of 64
  top 68
  trigonometry functions 66
  using 62
  west 68

Predefined operator
d (differential operator) 65
Preferences
graph 199
Prefix ops 228
Prefs menu 285
Arbitrary Constants 91
Auto Casing 32, 91
Auto Collapsar 83
Auto Simplify 91
Display Precision 57, 102
Graph 199
Show Palette 233
Principal value 7
Print command 10
Print monitor
adjusting 11
Printing
graphs 198
notebooks 10
page order 10
Print monitor 11
Problems
reporting 330
Programming graphs 209-220
Projectionist 299
Propositional schema 12
Propositions 25-48
assumptions 26, 28
available within theories 34
collapsing 48
comments 37, 225
conclusions 26, 28
connections 25
creating 26, 28
declarations 18
deleting 21
editing 237
exposing 48
formatting 47
getting information 25
in notebooks 17
logical relationship 27
main theory 25
moving 47
name declarations 39
organization 25
organizing 47
outlining 47
picture comments 37
rearranging 47
statements 26
text comments 37
types of 25
working statements 28

Q

Quadratic polynomials
solving with Isolate 94
Question mark 85
QuickDraw
vs PostScript 201
Quitting Theorist 19

R

radius
in Polar graphs 214
predefined constant 68
RAM
use of 4
Range op 80
entering 80
Ranges
creating 258
entering 258
of graph variables 213
Re
predefined function 64
Real
variable in graphs 173
Real numbers
arbitrary constants 91
as subscripts 74
declaring variables as 262
Rearranging
propositions 47

Red, Green, Blue color model 190
Reducing higher powers 31
Relational ops 72
  creating 258
  entering 73, 258
Replacement
  Substitute 98
  transformation rules 45
Reporting problems 330
Reserved words 62
Resizing
  comments 37
Resolution
  changing in 2D graphs 142
  changing in 3D graphs 164
Reversing
  Calculate 104
Revert command 19
RGB color model 190
right
  predefined constant 68
Rocket
  changing graph bounds 142, 164
  in 2D graphs 142
  in 3D graphs 164
Rocket icon 136
Rotating
  3D graphs 163
  animations 266
Round
  predefined function 64
Row Number command 268
RowsOf (matrix)
  predefined function 67
Rules and Declarations 20

S

Save as command 19
Save as Stationary command 19
Save command 19
Saving animations 195
Scientific notation
  entering 244
  example of 57
  using 57
Sec (Secant)
  predefined function 66
Sech (Hyperbolic secant)
  predefined function 66
Select In command 225, 233, 239
Select Out command 225, 233
Selecting
  Click 236
  Click-and-Drag 236
  Double-Click 236
  Double-Click-and Drag 236
  expressions 238
  matrices 239
  multiple objects 238
  N-ary expressions 239
  names 239
Selecting expressions 236
SemiLog graphs
  creating 154
  description of 138
  figure of 154
  plot proposition for 154
Settings
  display technology 199
  for Custom color 175
  for Solid color 176
Shade value range
  adjusting 185, 186
Shade values
  adjusting 184
  list of Complex 189
  list of Custom 188
  list of Gradient 186
Show Palette option 233
Showing propositions 48

Simplify 105-108
and Auto Simplify 108
and transformation rules 45, 121
conditional ops 80
effects on syntax 73
evaluate at ops 80
examples of 106, 107
integrals 78
partial derivatives 77
Simplify command 108
Simultaneous equation solving
in two variables 160
Sin (Sine)
predefined function 66
Sinh (Hyperbolic sine)
predefined function 66
Size box
figure of 135
in 2D graphs 143
in 3D graphs 165
Smallest number 57
Software
system 4
Solid color scheme 183
possible settings 176, 183
Solid colors
using 183
Solutions
multiple 7, 32
with arbitrary constants 92
Solving
with Isolate 94
south
predefined constant 68
Spacing of contours 150
Special cases 7
Special notebooks 19
Speed
animation playback 195
of processors 4
Spherical
functions 67
Spherical 3D command 219
Spherical 3D graphs
creating 178
description of 140
figure of 178
plot proposition for 178
Spherical contour plots 219
sqrt (square root) 75
Square root op 75
Square roots
entering 75
Standard Rules and Declarations 22
Startup
notebooks 19
Startup files
stationary files 19
Statement propositions 26
working 28
Statements
definition of 26
figure of 26
function 31
increment 30
maximum 30
minimum 30
order 31
parts of 26
types of 26
value 29
vs functions 73
working 28
Stationary files 19
as startup files 19
creating 20
names of 19
Stop Working Stmt command 28
Stopping working statements 28
Stretch to Fit (2D) 145
settings for 145
Stretch to Fit (3D) 167
Structure of expressions 224
Subscripted names 60

Subscripts
entering 74
for functions 61
for matrices 60
legal 60
log 61
meaning of 74
natural log 61
numerical 60
rounding to integers 60
Substitute 98-100
and wildcard variables 81, 100
figure of 98, 99
substitution equation 100
using 98, 263
vs Transform 120
vs transformation rules 46
Substitute command 99
Subtraction op 71
Summation op 78
and maximum statements 30
and minimum statements 30
Summations
entering 79
Expand 78
numerical 78
Surface plot propositions 181-192
figure of 183, 184, 186, 188, 190
introduction to 172
Surface plots
color scheme 172
color schemes 183-192
figure of 172
graphs that create 181
in 2D graph theories 156
introduction to 172
list of 181
location vector 172
Optical characteristics 172
Symbols
for names 59
System
minimum 4
System requirements 4

T

Tab key 226
Table of
Direct color models 192
operator hierarchy 227
ops 69
predefined behaviors 64
predefined names 64
wildcard variables 81
Tan (Tangent)
predefined function 66
Tanh (Hyperbolic tangent)
predefined function 66
Target
Substitute 98
Taylor expansions 122
Taylor series 122-124
creating 122
dialog box 123
example 33
expansion point 122
multidimensional 123
Taylor Series command 123
Text
copying expressions as 58
Text comment propositions 37
Theories
case 32
Theorist
class system 52
customizing 19, 20
fundamental objects 51
launching 19
Mac II version 4
non-procedural 72
notebooks 17
organizing principal 12
quitting 19
with other programs 10

Three-dimensional graphs 162-192
creating 171-180
figure of 135
introduction to 135
rotational freedom 163
vs two-dimensional graphs 163
Tips 261-268
ToCylindrical
predefined function 67
top
predefined constant 68
ToPolar
predefined function 67
ToSpherical
predefined function 67
Transform Rule command 46, 122
Transformation rules 45, 120
and wildcard variables 81, 120
creating 46
dialog box 121
example of 120
execute on Collect 45, 121
execute on Expand 45, 121
execute on Simplify 45, 121
execute on Transform 45, 121
figure of 45
invoking 45, 121
making recursive 122
pattern 45
replacement 45
scope 46
using 120
vs Substitute 46, 120
Translucent
optical characteristics 182
Transparent
optical characteristics 182
Trig Functions
distribution notebook 22
Trigonometry
functions 66
True Proportions (2D) 145
settings for 145
True Proportions (3D) 167
Truth
way to 13
Two dimensional Taylor series
and independence
declarations 43
Two-dimensional graph details
figure of 134
Two-dimensional graphs 141-158
creating 148-158
figure of 132
introduction to 132
vs three-dimensional graphs 163
Types of
class 52
expressions 51
names 39
propositions 25
statements 26
Typing in
example of 242

U

Uncalculate 104
examples of 105
Undefined
numbers 57
Undefined values 85
Underflow
value for 57
Undoing
Calculate 104
User defined names 41

Using
Apply 116, 117, 119
Calculate 101
Collect 112
color spectrum 184
Commute 96
Complex colors 188
Constants 53
Custom colors 186
D-Linear Operators 54
differentials 255, 256
Direct colors 190
distribution notebooks 21
evaluate at op 258
Expand 109
Expressionist 305
Functions 55, 73, 252
Gradient colors 184
independence declarations 43
index op 74
integrals 255
Integrate by Parts 125
Isolate 93
iterations 257
Knife 142, 164
Linear operators 253
M-Linear Operators 54
matrices 254
mouse with graphs 141, 162
Move Over 96
names 245
negation op 251, 262, 267
numbers 244
ops 69, 246
Orientation icon 165
parentheses 84
Projectionist 299
ranges 258
relational ops 258
resolution icons 142, 164
Rocket 142, 164
scientific notation 57
Solid colors 183
stationary files 20
subscripted variables 266
subscripts 74
Substitute 98
substitutions 263
the palette 231
Transform Rule 120
Variables 53
wildcard variables 81

V

Valid manipulations 7
Value statement 29
Variable dependence
dialog box 137
types of 137
Variable of integration 102
Variable palette 231
Variables
animation 193
as constants 262
as real numbers 262
class 53
complex 173
creating 53
declaring dependence for graphs 137
dependence 42
entering from palette 231, 232
examples of 53
in Contour plots 150
in graphs 213
in Line plots 149
in multiple independence declarations 44
independence 42
real 173
subscripted 266
using 53

Vectors
arguments 73
as subscripts 60
cross products 77
dot products 76
entering 76
function arguments 56
wildcard arguments 82
Viewport
figure of 132, 135
virtual size of (2D) 144
virtual size of (3D) 166
Viewport controls
definition of (2D) 145
definition of (3D) 167
figure of (2D) 144
figure of (3D) 166
Viewport cropping (2D) 145
Viewport cropping (3D) 167
Viewport lens (3D only) 167
Virginia
Custom color group 187

W

Warranty 331
west
predefined constant 68
Wide angle lens 167
Wildcard variables 81-82
and names 41
and Substitute 81, 100
and Transform 81
and Transform Rule 120
entering 81, 232
in function statements 31
table of 81
using 81
Working statements 28
automatic 28
creating 28
figure of 28
maximum 30
minimum 30
search for 28
to declare behavior 59
Types of 29
value 29
Wyoming
Custom color group 187

Y

YIQ color model 191

Z

Zero
division by 57, 72
Zero Contour graphs
creating 158
description of 139
figure of 158
plot proposition for 158